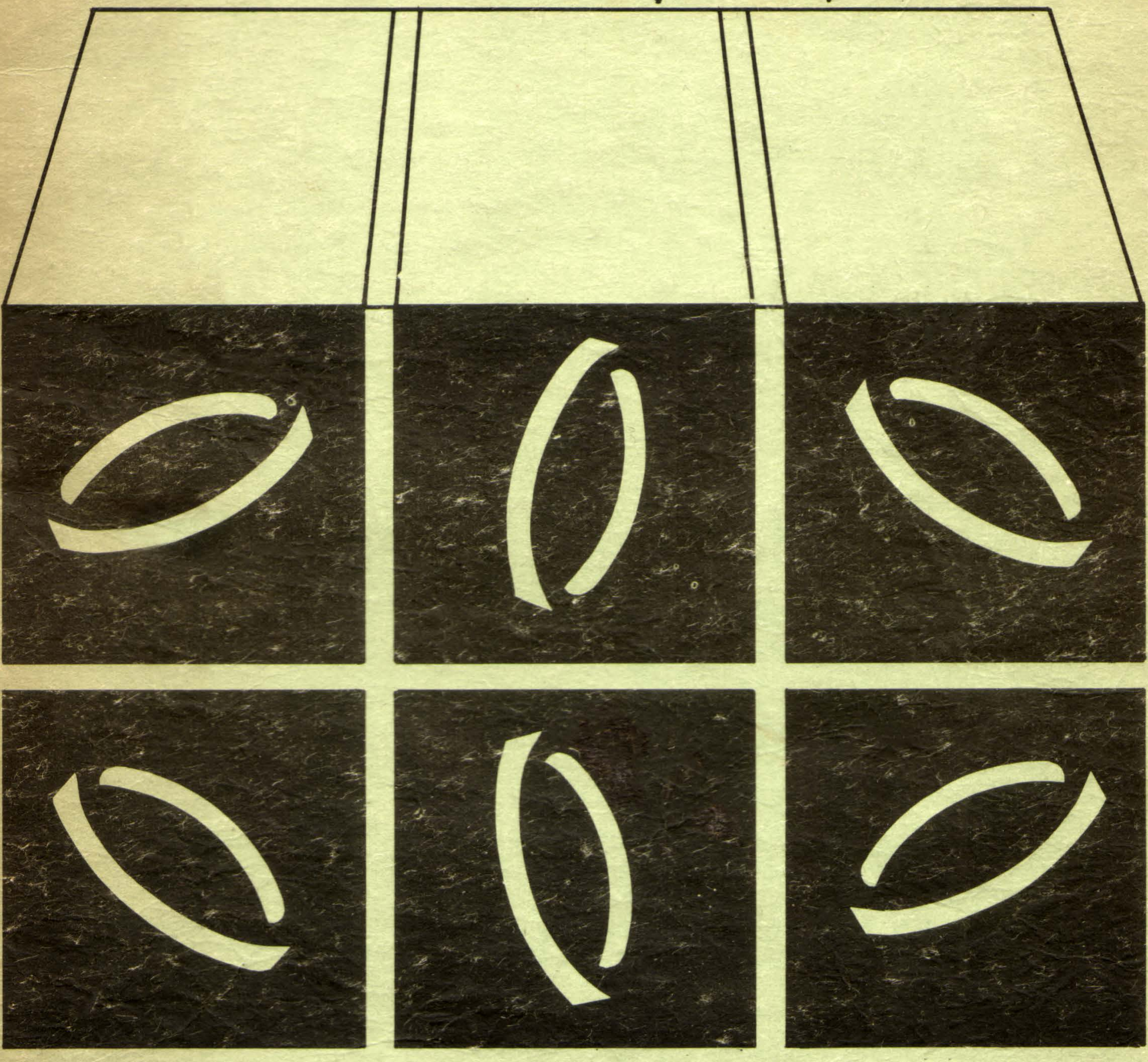
TIM S

timișoara
FMECTC

MICROCALCULATOR PERSONAL

MANUAL DE FUNCȚIONARE ȘI UTILIZARE



I.T.C.I. — F.M.E.C.T.C.
Bd. Gh. Lazăr nr. 9
Tel. : 3 00 78

TIM-S

MANUAL DE FUNCȚIONARE ȘI UTILIZARE

MICROCALCULATORUL TIM-S

A. MANUAL DE FUNCŢIONARE

1. **SCHEMA BLOC**

Microcalculatorul TIM-S, din punct de vedere tehnologic, este realizat pe o singură placă.

În figura 1 a şi 1 b este prezentată schema bloc constituită din :

1. MICROPROCESOR Z-80B (frecvenţa maximă de tact 6 MHz).

2. 16 Ko EPROM care conţin interpreterul BASIC realizat cu 8 circuite 2716.

3. 64 Ko RAM realizat cu 8 circuite 4164.

4. BLOCUL CIRCUITELOR DE INTRARE IEŞIRE realizat cu :

— 1×74LS174 ;

— 1×8225.

4.a. suplimentar pentru citirea tastaturii se folosesc şi 2×74LS32.

4.b. conversia TTL — semnale audio şi invers cu circuitul BM339.

4.c. convertorul TTL — semnal comandă difuzor constituit dintr-un tranzistor BC 107 şi rezistoare.

4.d. convertor TTL — RS 232 C realizat cu circuitele ROB 1488 şi ROB 1489.

4.e. Pentru comandarea unor imprimante paralele se mai folosesc cu rol de amplificatori-circuite CDB 404.

5. TAMPON VIDEO.

Pentru creşterea vitezei de calcul, memoria video a fost separată de memoria sistem prin trei registre (tip 8212) care memorează adresa şi datele care trebuie înscrise în memoria video. Acest mod de organizare permite şi lucrul sistemului la frecvenţe diferite fără o alterare a calităţii afişării.

6. BLOC CONTROL.

Acest bloc cu importanţă decisivă în buna funcţionare a sistemului, preia de pe magistralele sistemului şi cupla de extensie, toate informaţiile necesare (date, adrese, comenzi etc.) şi generează semnalele de comandă corespunzătoare (semnale de selecţie, comenzi de încărcare în tamponul video etc.). Tot acest bloc preia de la comutatorul extern comanda operatorului referitoare la tactul sistemului şi livrează procesorului tactul corespunzător.

7. OSCILATORUL CU QUARŢ (14 MHz).

Generează frecvenţa de 14 MHz necesară procesorului TV şi prin divizare se obţine tactul de 3,5 MHz, pentru Z-80.

8. OSCILATORUL CU QUARŢ (12 MHz).

Generează frecvenţa de 12 MHz din care se obţine tactul de 6 MHz pentru procesor.

9. CUPLA PENTRU EXTENSII.

Structura configuraţiei standard asigură un minim necesar de facilităţi pentru operator. În cazul în care se doreşte folosirea microcalculatorului TIM-S în aplicaţii mai complexe, se pot ataşa, prin intermediul acestei cuple module noi precum :

— interfaţa floppy-disc ;
— interfaţa plotter ;
— interfaţa reţea ;
— interfaţa CP/M etc.

La această cuplă sînt aduse o serie de semnale, care prin forţare la o anumită valoare logică, pot duce la o reconfigurare a sistemului, în sensul :

— schimbare adrese porturi ;
— schimbare adrese memorii ;
— deselectare porturi ;
— deselectare zone de memorie, etc.

La cuplă apar mai multe semnale care permit testarea şi diagnosticarea eventualelor defecte din sistem.

În partea de procesor TV există următoarele blocuri funcţionale :

10. DISPOZITIV DE COMANDĂ LOCALĂ.

Acest dispozitiv are drept scop :

— preluarea informaţiei din TAMPON VIDEO şi memorarea ei în memoria RAM (date şi informaţii de culoare, strălucire, clipire) şi memorarea lor în registre ;

— furnizarea semnalelor de stingere şi sincronizare pentru blocul video ;

— informarea BLOCULUI DE CONTROL asupra situaţiei din TAMPON VIDEO (dacă a fost sau nu preluată informaţia).

11. MEMORIA VIDEO 16 Ko RAM este realizată cu circuite 4116.

12. MULTIPLEXOR ADRESE realizat cu rezistoare.

Acest bloc permite adresarea memoriei fie de către microprocesor prin intermediul lui (5) şi (10) pentru scriere şi numai de către (10) pentru citire.

13. În acest registru se memorează temporar informaţii de culoare, strălucire, clipire pe cîte 8 pixeli (8212).

14. REGISTRU DE DATE.

Acest registru conţine datele ce urmează a fi afişate (74165+7474) în care scop încărcarea se face paralel şi descărcarea serie.

15. MIXER.

În acest bloc sînt amestecate informaţiile de culoare etc. cu informaţiile referitoare la date.

16. BLOCUL VIDEO.

Acestui bloc îi sînt aduse la intrări următoarele semnale :

— R ; G ; B, corespunzătoare culorilor ROŞU, VERDE, ALBASTRU ;

— BR ; caracter supraluminos ;

— H ; V, sincronizare pe linii şi cadre ;

— SH ; SV, stingere pe linii şi cadre.

În urma prelucrarii semnalelor mai sus pomenite, se obţin la ieşirea blocului :

— Semnalele R, G, B, şi SINCRO avînd amplitudinile şi impedanţele corespunzătoare pentru comanda unui monitor color prevăzut cu intrări de acest tip (de ex. „TELECOLOR 001").

— SVCC, Semnal Video Complex Color, compatibil, avînd amplitudinea şi impedanţa corespunzătoare comenzii unui monitor COLOR, prevăzut cu intrare de acest tip (de ex. „TELECOLOR 002").

Cu acest semnal poate fi comandat şi un monitor ALB/NEGRU (de ex. „TEHNOTON").

— TV, semnal standard de televiziune, un canal din banda III norma OIRT.

Cu acest semnal poate fi comandat, prin antenă orice tip de televizor COLOR sau ALB/NEGRU.

NOTA 1. După caz în prospectul care va însoţi microcalculatorul, se va specifica sistemul de CODARE-COLOR cu care este codat produsul.

II. DESCRIEREA FUNCŢIONALĂ GENERALĂ

BLOCUL DE CONTROL

Pentru a putea lucra cu memorii de tip PROM (EPROM) ieftine deci cu timp de acces mare, după RESET sau pornirea sursei, sistemul este obligat să lucreze pe frecvenţa de 3,5 MHz şi conţinutul PROM-urilor se transferă în RAM (care au un timp de acces convenabil) după care PROM-urile se dezactivează şi sistemul lucrează numai cu RAM. Acest lucru permite unui utilizator avizat să modifice în continuare interpreterul BASIC conform nevoilor sale (ex. schimbarea subrutinelor pentru imprimanta MIM-40 cu cele pentru SCAMP 9335).

Ţinînd cont că interpreterul BASIC se găseşte în RAM cu ajutorul unui bistabil (comandabil soft) se interzise hard scrierea în primii 16 Ko de RAM (pentru evitarea accidentelor de operare). Tot în scopul asigurării posibiltăţii utilizării unor circuite mai lente, în timpul unor operaţii de I/O procesorul lucrează pe tactul de 3,5 MHz indiferent de poziţia comutatorului. Frecvenţa pe care lucrează sistemul se poate forţa soft (prioritar comutatorului) printr-un bit din portul C al circuitului 8255. Tot din acest bloc se generează semnale de RAS, CAS, MUX şi WR necesare memoriilor dinamice de 64 Ko conform diagramei din fig. 2.1. Ca observaţie, folosind acest tip de generare a semnalului CAS (neconcomitent cu WR negat) permite cîştigarea cîtorva nsec la adresarea RAM-urilor.

În blocul de control se obţin semnalele de selecţie pentru memorii şi circuite de I/O. În principiu selecţia se realizează astfel : atît memoriile cît şi circuitele de I/O se menţin selectate, cu excepţia momentelor cînd este obligatoriu să fie deselectate (deci ele se deselectează la nevoie şi nu folosind metoda clasică în care se selectează la nevoie). Deşi s-ar părea că apare conflict pe magistrala de date, acesta nu există, deoarece pe perioada cînd atît MREQ negat cît şi IOREQ negat sînt inactive (deci memoria şi circuitele I/O sînt selectate simultan) nu apare semnalul RD negat.

Acest mod de selecţie are următoarele avantaje :

— adresarea unui circuit începe în T1 (o dată cu stabilirea adreselor) ;

— permite folosirea circuitelor din familia I 8080 împreună cu Z-80 fără a mai fi necesară prelucrarea semnalelor RD negat, respectiv WR negat ;

— folosirea de circuite mai lente fără introducere de stări de aşteptare la frecvenţa de tact de 6 MHz.

DIALOGUL MICROSISTEM-PROCESOR VIDEO

Procesorul video are ca tact de bază tactul de 7 MHz (X1). Prin divizare se obţine X4 (f×4=0,875 MHz), vezi fig. 2.2. Acest tact determină momentele cînd se fac scrieri în memoria video X4=0, respectiv se citesc octeţi de date şi atribute. În momentul cînd sistemul face o tentativă de scriere în tamponul video (la adrese 4000H—6000H) se testează dacă din tamponul video s-a scris informaţia în memoria video (OP=0) şi se înscrie noua informaţie cu frontul crescător al lui WR negat. Dacă OP=1 se suspendă tactul sistemului în T3 pînă la golirea tamponului, după care se încarcă noua informaţie tot cu frontul crescător al lui WR negat.

Dacă sistemul funcţionează cu tact de 3,5 MHz (T=1141 ns) se poate demonstra că nu apar astfel de stări de aşteptare decît în cazul unor instrucţii care la interval de 3 tacturi de procesor fac scrieri în această zonă de memorie (PUSH, CALL etc.). În partea de procesor video bistabilul OIS (1) semnalizează că are loc o operaţie de scriere în RAM şi resetarea acestuia determină resetarea lui OP.

TASTATURA

Tastatura este organizată ca o reţea de 5×8 bare, la fiecare intersecţie se poate face un contact electric apăsînd tasta dorită. Fiecare linie este selectată de către o linie de

adrese (A8-A15) prin intermediul unei porți SAU (74LS432). Cele 5 coloane care normal au nivel logic "1" (prin rezistoare de 10 kohmi legate la +5 V) sînt citite prin portul A al circuitului 8255 (PA0-PA4).

Scanarea tastaturii se face sub interpreterul BASIC, la fiecare 10 msec. (întreruperi generate de sincro cadre din procesor video). Există totuși următoarele excepții : se testează direct tasta CAPS SHIFT și BREAK, atunci cînd numai ele interesează (subrutinele de LOAD, SAVE etc.).

Scanarea tastaturii se bazează pe următoarele facilități oferite de Z-80 :

— la IN A, OUT A, conținutul acumulatorului (din ciclul M1) apare pe liniile de adrese A8-A15 în tot timpul execuției instrucției ;

— la instrucțiile IN și OUT care folosesc pentru adresare registrul C, conținutul registrului B apare pe liniile de adresă A8-A15.

Ținînd seama de aceasta orice utilizator poate să-și facă propria rutină de citire a tastaturii și să dezactiveze eventual întreruperile (aceasta duce la o sporire a vitezei de calcul atunci cînd nu ne interesează toate tastele doar la momente bine precizate de timp).

Dacă se dorește citirea întregii tastaturi propunem următoarea soluție :

```
DI
.
.
RST 38 (sau CALL 02BF)
```

după care dacă a fost apăsată o tastă, codul ei se găsește la adresa 23560 (5CO8) și la adresa 23611 (5C3B) se găsește informația referitoare la faptul că a fost sau nu o tastă.

CUPLA PENTRU EXTENSII

Cupla pentru extensii permite legarea microsistemului TIM-S cu :

— interfețe pentru operații de I/O ;

— testoare ;

— extensii care să asigure : rulare de programe sub alte sisteme de operare (de exemplu CP/M), completare interpretor BASIC existent (rețea locală, floppy-disk etc.).

Lista semnalelor la această cuplă este dată în fig. 2.4. Următoarele semnale provin direct de la microprocesor și permit o încărcare de o sarcină TTL LS :

— A0-A15 ;
— D0-D7 ;
— BUSRQ negat ;
— RFSH negat ;
— WR negat ;
— BUSACK negat ;
— MREQ negat ;
— HALT negat ;
— NMI negat ;
— RD negat ;
— M1 negat ;
— CLK ;
— RESET negat ;
— IORQ negat ;
— INT negat.

Semnalele de mai jos au următorul rol :

— Op — octet prezent, indică prezența unui octet și a adreselor corespunzătoare în tampoanele video ; în scopul testării acest semnal se poate forța ("0" sau "1").

— TVD negat — TP dezactivat, prin forțare la "1" se inhibă scrierea în tampoanele video (facilitate oferită pentru ca TIM-S să poată lucra și sub alte sisteme de operare).

— ROP — reset OP, se poate atît citi cît și forța pe valoarea dorită.

— ROM — indică faptul că TIM-S lucrează cu sistemul de operare în ROM și se poate forța la valoarea dorită.

— ROMCS negat indică faptul că adresa din ciclul de memorie este cuprinsă între 0000H-3FFFH și sistemul lucrează cu ROM.

— ROMD negat — ROM dezactivat. Un semnal "0" pe acest pin dezactivează ROM-ul intern.

— RAMD negat — RAM dezactivat. Un semnal "0" pe acest pin dezactivează RAM-ul intern.

— C5-bit PC5 din 8255.

— DA0-DA7 — magistrala de date separată de D0-D7 a microprocesorului prin rezistente (acest lucru permite testarea rapidă a microcalculatorului TIM-S prin analiza de semnături).

— CP — la citire indică pe ce tact lucrează microprocesorul. Se poate forța ('0' — 3,5 MHz, "1" — 6 MHz).

ALTE FACILITĂȚI HARD

Patru biți de PC ai circuitului 8255 sînt folosiți în comanda funcționării sistemului și anume :

— PC1-C1 — împiedică scrierea în primii 16 Ko de RAM (C1=0).

— PC2-C2 — împiedică înscrierea în procesorul video (C2=1).

— PC3-C3 — forțează tactul microsistemului pe frecvența de 3,5 MHz prioritar față de poziția comutatorului (C3=0).

— PC4-C4 — dezactivează orice adresare a RAM-ului sistem (C4=0).

Portul PA al circuitului 8255 se poate adresa atît cu FEH cît și cu F9H (numai pentru citire).

Porturile de mai jos au următoarele adrese :

— PA-E0H, FEH ;
— PB-E2H ;
— PC-E4H ;
— RC-E6H — registru de comandă ;
— 74LS174-FEH (numai scriere).

III. INTERFAȚA SERIE ȘI PARALELĂ

Microcalculatorul TIM-S în configurația de bază (fără extensii) este prevăzut cu o interfață de tip serie și una de tip paralel pentru acționarea imprimantelor. Programele de comandă a imprimantelor sînt incluse în sistemul de operare rezident în EPROM.

Microcalculatorul TIM-S în comanda imprimantelor face uz de conceptul de canal. Un ca-

nal este operaţional, adică acceptă date pentru a fi trimise la o imprimantă, numai dacă se găseşte în starea "deschis". Un canal în starea închis nu acceptă date, rezultînd un Raport de eroare. Utilizatorul în limbajul BASIC TIM-S dispune de 12 canale şi poate manipula starea canalelor prin comenzile OPEN şi CLOSE.

Comanda OPEN # N, "nume" deschide canalul cu numărul N (N=4 . . . 15). Prin "nume" utilizatorul stabileşte tipul canalului în sensul că indică modul de transmitere a datelor (serie, paralel, cod ASCII, octet în paralel) în cadrul unui anume protocol. Comanda CLOSE N are ca efect închiderea canalului cu numărul N (N=1 . . . 15).

La microcalculatorul TIM-S se pot conecta următoarele tipuri de imprimante :

1. Miniimprimanta MIM-40.
2. Imprimanta SCAMP (CDC 9335) în regimurile :
— paralel ;
— serie ;
— grafic.
3. Imprimanta matricială paralelă DZM-180.
4. Imprimanta ROBOTRON 6311 în regimul :
— serie, paralel.
5. Imprimanta matricială paralelă ROBOTRON 1152.
6. Consola CENTRONIX-serială.
7. Imprimanta paralelă rapidă VIDEOTON ES184 (800 linii/minut).

La microcalculatorul TIM-S tipurile de canale şi funcţiile atribuite acestora sînt :

"nume"	tip canal	funcţia atribuită ;
G	grafic	transferă conţinutul memoriei ecran (4000-57FF) în paralel, octet după octet la interfaţa paralelă ;
g	grafic	transferă conţinutul memoriei tampon (5B00-5BFF) în paralel octet după octet ;
A	asincron	transferă conţinutul memoriei ecran (4000-57FF) în cod ASCII la interfaţa serie în formatul cu 7 cifre binare şi 2 biţi de stop fără paritate în protocol BUFFER-BUSY (DTR) viteza 300, 1 200 sau 2 400 bd ;
a	asincron	transferă conţinutul memoriei tampon (5B00-5BFF) în cod ASCII la interfaţa serie în formatul cu 7 cifre şi 2 biţi de stop fără paritate în protocol. BUFFER-BUSY, viteza 300, 1 200, 2 400 bd ;
M	paralel	transferă conţinutul memoriei ecran (4000-57FF) la interfaţa paralelă, cu respectarea cerinţelor de comandă impuse de miniimprimantă MIM-40 ;
m	paralel	transferă conţinutul memoriei tampon (5B00-5BFF) la interfaţa paralelă cu respectarea cerinţelor de comandă impuse de miniimprimantă MIM-40 ;
B	paralel	transferă conţinutul memoriei ecran (4000-57FF) la interfaţa paralelă, în cod ASCII cu 7 cifre binare ; cu dialogul STROB-BUSY ;
b	paralel	transferă conţinutul memoriei tampon (5B00-5BFF) la interfaţa paralelă în cod ASCII cu 7 cifre binare, cu dialogul STROB-BUSY ;
X	paralel	transferă conţinutul memoriei ecran (4000-57FF) la interfaţa paralelă în cod ASCII cu 7 cifre binare, cu dialogul RUF-END ;
x	paralel	transferă conţinutul memoriei tampon (5BOO-5BFF), la interfaţa paralelă, în cod ASCII cu 7 cifre binare, cu dialogul RUF-END.

La indicarea unui număr de canal N mai mare decît 15 sau a unui nume altul decît G, g, A, a, M, m, B, b, X sau x rezultă ca mesaj un Raport de eroare.

După iniţializarea microcalculatorului TIM-S (la punere sub tensiune sau la acţionarea butonului RESET) canalele 4—15 sînt declarate închise. Fără a deschide în mod explicit unul din aceste canale, utilizatorul dispune de comanda COPY care are ca efect transferul conţinutului memoriei ecran în paralel la interfaţa paralelă — regim grafic.

Acelaşi rezultat se obţine prin :
OPEN # N, "G"
. . . .
PRINT # N
unde N=4 . . . 15

Se menţionează de asemenea că după iniţializarea microcalculatorului TIM-S în cadrul sistemului de operare se deschide implicit canalul 3 ca fiind de tipul B. Comenzile LPRINT şi LLIST în sintaxa fără N şi „nume" sînt asociate canalului 3 şi în consecinţă vor conduce la comanda imprimantei SCAMP în regim paralel.

Facilitățile oferite utilizatorului sînt puse în evidență prin cîteva exemple :

Exemplul 1 :

```
OPEN # 4, "G"
. . . . . . . . . . → instrucțiuni care afișează
                      informația alfanumerică
                      și grafică pe ecran
PRINT # 4
CLOSE # 4
```

Programul va realiza copie de ecran la imprimanta CDC 9335, în regimul grafic prin interfața paralelă.

Exemplul 2 :

```
OPEN # 6, "a"
(PRINT) LPRINT # 6, "text"
CLOSE # 6
```

Programul va realiza tipărirea unui text la una din imprimantele CDC 9335, ROBOTRON K6311 sau la consola CENTRONIX, în regimul serie, prin interfața serie.

Exemplul 3 :

```
OPEN # 7, "M"
. . . . . . . . . . → instrucțiuni care afișează
                      informația alfanumerică
                      și grafică pe ecran
PRINT # 7
CLOSE # 7
```

Programul va realiza copie de ecran la mini-imprimanta MIM-40, prin interfața paralelă.

Exemplul 4 :

```
OPEN # 8, "b"
(PRINT) LPRINT # 8, "text"
CLOSE # 8
```

Programul va realiza tipărirea unui text la imprimanta CDC 9335, în regim paralel, prin interfața paralelă.

Exemplul 5 :

```
OPEN # 9, "X"
. . . . . . . . . . → instrucțiuni care afișează
                      informația numerică pe
                      ecran
PRINT # 9
CLOSE # 9
```

Programul va realiza copie de ecran la una din imprimantele DZM 180 sau VIDEOTON ES7184, în regim paralel, prin interfața paralelă.

Exemplul 6 :

```
OPEN # 10, "A"
. . . . . . . . . . → instrucțiuni care afișează
                      informația alfanumerică
                      pe ecran
PRINT # 10
CLOSE # 10
OPEN # 11, "b"
(PRINT) LPRINT # 11, "text"
(LIST) LLIST # 11
CLOSE # 11
```

Programul va realiza copie de ecran la una din imprimantele CDC 9335, ROBOTRON K6311 sau la consola CENTRONIX, în regim serie, prin interfața serie apoi va realiza tipărirea unui mesaj la imprimanta CDC 9335, în regim paralel, prin interfața paralelă, iar apoi va realiza listarea programului BASIC la imprimanta CDC 9335, în regimul paralel, prin interfața paralelă.

Exemplul 7 :

```
OPEN # 12, "a"
. . . . . . . . . . .
(LIST) LLIST # 12
CLOSE # 12
OPEN # 13, "x"
(PRINT) LPRINT # 13, "text"
CLOSE # 13
```

Programul va realiza listarea programului la una din imprimatele CDC 9335, ROBOTROM K6311 sau la consola CENTRONIX, în regim serie, prin interfața serie, apoi va realiza tipărirea unui text la una din imprimatele DZM 180 sau VIDEOTON ES7184 în regim paralel, prin interfața paralelă.

Exemplul 8 :

```
OPEN # 4, "a"
OPEN # 5, "x"
OPEN # 6, "A"
(PRINT) LPRINT # 4, "text 1"
(PRINT) LPRINT # 5, "text 2"
. . . . . . . . . . → instrucțiuni care afișează
                      informația alfanumerică
                      pe ecran
PRINT # 6
CLOSE # 4
CLOSE # 5
CLOSE # 6
```

Programul va realiza tipărirea mesajului "text 1" la una din imprimantele ROBOTRON K6311, CDC 9335 sau la consola CENTRONIX în regim serie, prin interfața serie, tipărirea mesajului "text 2" la una din imprimantele DZM 184, sau VIDEOTON ES7184, în regim paralel, prin interfața paralelă, copie pe ecran corespunzător primului grup de instrucțiuni care afișează informația alfanumerică pe ecran, la una din imprimantele CDC 9335, ROBOTRON K6311 sau la consola CENTRONIX în regim serie, prin interfața serie.

MESAJE DE EROARE

Invalid file name — dacă se dă un nume de canal diferit de cele specificate.

Invalid stream — dacă se apelează un canal care nu s-a deschis anterior.

Exemplul 9 :

```
E1 OPEN # 4, "b"
E2          I
.           I → linii de
.           I   program BASIC
.           I
EN (LIST) LLIST # 4, E2
```

Programul va realiza listarea programului BASIC, începînd cu linia cu numărul E2, la imprimanta CDC 9335, în regim paralel, prin interfața paralelă.

Exemplul 10 :

OPEN # 4, "nume"

(PRINT) LPRINT # 4, "text"

Programul va da mesajul de eroare : „Invalid file name" dacă "nume" este o literă diferită de cele cunoscute (M, m, G, g, A, a, B, b, X, x).

Exemplul 11 :

OPEN # 17, "a"

(PRINT) LPRINT # 4, "text"

Programul va da mesajul de eroare, „Invalid stream" dacă se apelează un canal cu numărul în afara domeniului (4 ... 15).

Același mesaj de eroare apare și dacă se comandă tipărirea folosind un canal care nu a fost deschis anterior comenzii.

PROGRAMAREA IMPRIMANTEI SCAMP

Imprimanta SCAMP permite un mod de lucru prin programare. Recunoaște un set de c.a. 30 de coduri de comandă (ASCII). Pentru trimiterea acestor coduri din program BASIC, ținînd cont de dialogul purtat între imprimantă și calculator, la trimiterea datelor, cît și de existența unor inversoare la ieșirile portului B din circuitul de interfață 8255, se pot citi 3 instrucțiuni OUT pentru fiecare octet de trimis, în următorul fel :

e1	OUT adr. port	— inversul octetului format din cei 7 biți de date și bitul de comandă inactiv
e2	OUT adr. port	— inversul octetului format din cei 7 biți de date și bitul de comandă activ
e3	OUT adr. port	— inversul octetului format din cei 7 biți de date și bitul de comandă inactiv

Ex : Pentru trimiterea codului SO (EO în ASCII) — care are drept consecință tipărirea primului rînd de tipărit cu caractere dublu late — se calculează informația pentru trimiterea codului ca în tabelul de mai jos :

Nr.	Octetul format din date și bit de c-dă	Hexa	Inversul octetului	Hexa	Zecimal
1	1 0 0 0 1 1 1 0	8E	0 1 1 1 0 0 0 1	71	113
2	0 0 0 0 1 1 1 0	0E	1 1 1 1 0 0 0 1	F1	214
3	1 0 0 0 1 1 1 0	8E	0 1 1 1 0 0 0 1	71	113

Cunoscînd că adresa portului 3 al circuitului 8255 este E2 pentru trimiterea codului de comandă SO se folosesc următoarele 3 instrucțiuni :

e1 OUT 226, 113

e2 OUT 226, 241

e3 OUT 226, 113

În tabelul de mai jos se dau gata calculate aceste valori pentru coduri de comandă specificate în manualul de utilizare al imprimantei SCAMP. Utilizatorului îi rămîne să insereze în program OUT-uri succesive folosind valorile calculate în tabel :

ASCII	HEXA	A	B	C	D	E	F	G	H	I
CR	OD	120	248	120	—	—	—	—	—	—
LF	OA	117	245	117	—	—	—	—	—	—
FF	OC	115	243	115	—	—	—	—	—	—
SO	OE	113	241	113	—	—	—	—	—	—
SI	OF	112	240	112	—	—	—	—	—	—
ESC 6	1B 36	100	228	100	73	201	73	—	—	—
ESC 7	1B 37	100	228	100	72	200	72	—	—	—
ESC 8	1B 38	100	228	100	71	199	71	—	—	—
ESC 9	1B 39	100	228	100	70	198	70	—	—	—
ESCSP	1B 20	100	228	100	95	223	95	—	—	—
ESC 1	1B 21	100	228	100	94	222	94	—	—	—
ESC ′	1B 27	100	228	100	82	209	82	—	—	—
ESC 0	1B 30	100	228	100	79	206	79	—	—	—
ESC 1	1B 31	100	228	100	78	205	78	—	—	—
ESC B	1B 42	100	228	100	61	189	61	—	—	—
ESC C	1B 43	100	228	100	60	188	60	—	—	—
ESC e	1B 40	100	228	100	63	191	63	—	—	—
ESC A	1B 41	100	228	100	62	192	62	—	—	—
ESC n	1B6031	100	228	100	31	159	31	78	206	78
—	1B6032	100	228	100	31	159	31	77	205	77
n=1,4	1B6033	100	228	100	31	159	31	76	204	76
	1B6034	100	228	100	31	159	31	75	203	75
ESC 4	1B 34	100	228	100	75	203	75	—	—	—
ESC 5	1B 35	100	228	100	74	202	74	—	—	—
RS	1E	97	225	97	—	—	—	—	—	—
VT	OB	116	224	116	—	—	—	—	—	—
DLE	10	111	239	111	—	—	—	—	—	—

ASCII	HEXA		A	B	C	D	E	F	G	H	I
DC2	12		109	237	109	—	—	—	—	—	—
DC4	14		107	235	107	—	—	—	—	—	—
NAC	15		106	234	106	—	—	—	—	—	—
SYN	16		105	233	105	—	—	—	—	—	—
ETB	17		104	232	104	—	—	—	—	—	—
CAN	18		103	231	103	—	—	—	—	—	—
EM	19		102	230	102	—	—	—	—	—	—
SUB	1A		101	229	101	—	—	—	—	—	—
DC1			110	238	110	—	—	—	—	—	—
DC3	13		108	236	108	—	—	—	—	—	—
ESC D	1B	44	100	228	100	59	187	59	—	—	—
ESC L	1B	4C	100	228	100	51	179	51	—	—	—
ESC G	1B	47	100	228	100	56	184	56	—	—	—
ESC H	1B	48	100	288	100	55	183	55	—	—	—
ESC I	1B	49	100	228	100	54	182	54	—	—	—
ESC R	1B	52	100	228	100	45	173	45	—	—	—
BEL	07		120	248	120	—	—	—	—	—	—

Se recomandă păstrarea pe casetă sub formă de subprograme a instrucţiunilor necesare pentru trimiterea fiecărui cod în parte. Apelînd la aceste subprograme, trimiterea unui cod de comandă spre imprimata SCAMP, apoi la o instrucţiune de apel subprogram : e GO SUB adr. progr.

Utilizatorul odată familiarizat cu modul de trimitere a unor coduri de comandă, poate introduce un set de caractere proprii, cu caractere generate pe o matrice 9×9. Odată încărcat acest set de caractere cu codul de comandă ESCL, cu octeţii de informaţie trimişi în acelaşi mod spre imprimată ca şi octeţii de comandă, prin codurile ESCD, respectiv ESCR, se poate selecta, respectiv deselecta tipărirea setului de caractere definit de utilizator în locul celui standard. Amănunte privind modul de încărcare a setului nou de caractere, precum şi modul de generare a noilor caractere se pot urmări în manualul de utilizare al imprimatei SCAMP.

Cupla de extensie (circuit imprimat 961-431-060)

	A	MIJLOC (B)	C
32	—5 V	—5 V	—5 V
31	+12 V	+12 V	+12 V
30	A12		A11
29	A14		A13
28	A5		A6
27	A2		RD negat
26	A1		AO
25	OP		M1 negat
24	TUD negat		ROMCS negat
23	ROP		ROMD negat
22	A10		IORQ negat
21	A9		CP
20	A7		A8
19	BUSRQ negat		RESET negat
18	D5		DO
17	D7		IORQ negat
16	ROM		INT negat
15	RFSH negat		D1
14	WR negat		OS
13	A3		RAMD negat
12	BUSAK negat		A4
11	MREQ negat		WAIT negat
10	HALT negat		DA7
9	NMI negat		DAQ
8	D2		CLK
7	D3		D6
6	A15		D4
5	DA2		DA5
4	DA3		DA6
3	DA4		DA1
2	+5 V	+5 V	+5 V
1	GND	GND	GND

Fig. 2.4.

Cupla de extensie (circuit imprimat 961-431-060 REVA)

	A	MIJLOC (B)	C
32	—5 V	—5 V	—5 V
31	+12 V	+12 V	+12 V
30	A12	GND	A11
29	A14	GND	A13
28	A5	GND	A6
27	A2	GND	RD negat
26	A1	GND	AO
25	OP	GND	M1 negat
24	THD negat	GND	ROMCS negat
23	ROP	GND	ROMD negat
22	A10	GND	IORQD negat
21	A9	GND	CP
20	A7	GND	A8
19	BURSQ negat	GND	RESET negat
18	D5	GND	DO
17	D7	GND	IORQ negat
16	ROM	GND	INT negat
15	RFSH negat	GND	D1
14	WR negat	GND	OS
13	A3	GND	RAMD negat
12	BUSAK negat	GND	A4
11	MREQ negat	GND	WAIT negat
10	HALT negat	GND	DA7
9	NMI negat	GND	DAO
8	D2	GND	CLK
7	D3	GND	D6
6	A15	GND	D4
5	DA2	GND	DA5
4	DA3	GND	DA6
3	DA4	GND	DA1
2	GND	GND	GND
1	+5 V	+5 V	+5 V

Fig. 2.4.

IV. BLOCURI FUNCȚIONALE ALE CALCULATORULUI

1. PROCESORUL VIDEO

Este alcătuit din :

— 16 Ko de memorie tip 4116 din care 6 Ko sînt folosiți pentru memorarea datelor ce trebuiesc afișate și următorii aprox. 300 H octeți pentru atribute :

— tamponul video format din circuitele 8212a, 8212b — unde se încarcă adresa și 8212c — unde se încarcă datele în momentul în care procesorul face o scriere în memoria video ;

— multiplexorul de adrese pentru memoria video format din circuitele : 74157a și 74157b care multiplexează adresele de linii și coloane, circuitul 74157d cu rol în multiplicarea accesului la date sau atribute și rezistențele R60—R27 cu rol în multiplexarea adreselor pentru accesul la scriere sau accesul la video la citire ;

— circuitul 8212d folosit ca registru pentru atribute ;

— circuitul 74165 și bistabilele 74g — pentru încărcarea paralelă și serializarea datelor ;

— multiplexorul 74157c pentru multiplexarea informației de PAPER și INK funcție de logică de semnalizare a datelor (mixer video) ;

— dispozitivul de comandă locală a memoriei care pe lîngă semnalele VRAS negat, VMUX negat și VCAS negat mai generează și semnale de încărcare atribute LOADA ; date LOADD și semnalele de dialog cu microprocesorul cînd acesta face scrierea în memoria video (OIS negat, ROP negat) ;

— circuitele tip 74161 care împreună cu logira aferentă asigură controlul adreselor multiplexate prin rircuitele 74157a, b spre RAM video, a semnalelor de sincronizare și stingere pentru linii și cadre necesare ieșirilor pe televizor sau monitor.

Afișarea alfanumerică pe ecran se face prin construirea unni caracter cu o matrice 8 8 în care fiecare bit constituie un pixel. Valoarea logică a fiecărui bit condiționează la ieșirile multiplexorul 74157c informația RGB corespunzătoare PAPER-ului pentru bit O și INK-ului pentru bit 1. Fiecare matrice 8 8 are un octet de atribut cu următoarele semnificații :

D 7	D 6	D 5	D 4	D 3	D 2	D 1	D 0
FLASH	BRIGHT		RGB PAPER			RGB INK	

În modul de afişare 8 * 8 biţi pe caracter se pot afişa 32 caractere/rînd şi 24 de rînduri :

```
                        1 CHAR                                              2 CHAR
RÎND A                  0  0  0  0  0  0  0  0                (A+1)   0  1  0
  1     (A+1 * 256)     0  0  0  0  1  0  0  0                        1
        (A+2 * 256)     0  0  1  0  0  1  0  0                        0
                        0  0  1  0  0  1  0  0                        0
                        0  0  1  1  1  1  0  0
                        0  0  1  0  0  1  0  0
                        0  0  1  0  0  1  0  0
        (A+7 * 256)     0  0  1  0  0  1  0  0
-----------------------------------------------------------------------------------
RÎND    (A+32)             0  0
  2                        1
                           1
                                  33 CHAR
-----------------------------------------------------------------------------------
```

Datele se afişează pe ecran sub forma a 3 zone orizontale ; fiecare zonă are 8 rînduri ; fiecare rînd are 8 linii. Adresele de unde sînt citite datele pentru afişare sînt în cadrul unei zone conform figurii de mai sus. Adresele de început de zonă corespunzătoare celor 3 zone sînt :

— zona 1 — A=4000H
— zona 2 — A=4800H
— zona 3 — A=5000H

Atributele se găsesc în memorie în spaţiul 5800-5AFF, atributul corespunzător caracterului din colţul stîng sus al ecranului se găseşte la 5800.

FOARTE IMPORTANT : afişarea pe ecran foloseşte numai jumătate din memoria video dacă această jumătate de memorie nu este bună se poate încerca cealaltă jumătate prin schimbarea pinului 11 de la 74157b de la masă la +5 V.

Memoria video dublează practic o zonă de aceeaşi lungime din memoria 4164 a sistemului.

Cînd microprocesorul scrie ceva în memoria video, face simultan o scriere şi în zona corespunzătoare a memoriei 4164 şi o încărcare în tampoanele 8212a, b, c a adresei şi datelor, pe care procesorul video le va prelua cînd va trece în regim acces procesor. Semnalele de comandă pentru memorie, semnalele de dialog cu microprocesorul şi semnalele de încărcare atribuite şi date sînt conform fig. 2.2. a.

Încărcarea în tampoane se face prin semnalul OP negat aplicat la pinul 11. Semnalul OP negat se poate forţa din cuplă la capătul rezistenţei R100. De asemenea semnalul ROP negat se poate forţa din cuplă la capătul rezistenţei R46. Prin acţionarea butonului RESET se poate simula o scriere în RAM-ul video prin activarea (în 0) pinului 13 de la 74d.

Ieşirile x5—x9 de la numărătoarele 74161 reprezintă adresa caracterului de afişat în cadrul rîndului, iar y0—y7 numărul liniei TV în curs de afişare. Informaţia utilă se afişează pe ecran într-o fereastră centrală dreptunghiulară delimitată de un chenar numit BORDER, controlul BORDER-ului se face prin semnalul BORD de la ieşirea 6 a bistabilului 74f. BORD=1 pentru fereastră. BORDER-ul poate primi 8 culori selectabile printr-un OUT FE, în acumulator (înainte de OUT) completînd informaţia RGB pentru culoarea chenarului în primii 3 biţi. Rezistenţele R80, R81, R82 au rol de a transmite spre RGB extern informaţia RGB corespunzătoare BORDER-ului de la ieşirile circuitului 74LS174, în momentul cînd afişarea liniei TV se face pe BORDER. În fig. 4.1 se prezintă diagrama de semnale pentru o linie TV de la mijlocul ecranului.

2. GENERATORUL DE TACT

În sistem există 2 oscilatoare cu cuarţ formate din porţile din circuitul 404c care furnizează frecvenţele de :

— 12 MHz pentru obţinerea tactului de 6 MHz pentru microprocesor ;
— 14 MHz pentru obţinerea tactului de 7 MHz, de 3,5 MHz şi tactului pentru procesorul video. Tactul de 14 MHz este divizat cu ajutorul circuitului 4193a.

Deoarece sistemul funcţionează la frecvenţă mare (6 MHz) se impune ca tactul la procesor să aibă :

— fronturi$<=$20 ns ;
— V/OH$>=$4,4 V ;
— V/OL$<$0,4 V.

În acest scop se foloseşte un montaj cu tranzistor 2N709.

Bistabilele 474b, 474a, împreună cu porţile din jurul lor asigură :

— comutarea tactului de 3,5 MHz/6 MHz ;
— suspendarea tactului microprocesorului în cazul apariţiei unui conflict microprocesor—procesor video.

COMUTAREA TACTULUI

Cererea de comutare tact apare pe pinul 2 al circuitului 474b din 3 surse :

— bistabilul ROM (474c) — pentru a putea lucra cu EPROM-uri lente se forţează tactul pe 3,5 MHz indiferent de poziţia comutator extern ;

— bitul PC3 de la circuitul 8255 ;

— prin comutator extern.

În timpul utilizării calculatorului apar situații în care este nevoie să forțăm soft tactul de 3,5 MHz.

— prin semnalul CP de la cupla sistem. Acest semnal se forțează cu o poartă colector în gol sau TS de către o eventuală extensie legată la cupla de magistrală externă.

Diagramele corespunzătoare acestui bloc funcțional apar în fig. 4.2. (pentru suspendare tact).

Schimbarea tactului se realizează astfel :

— bistabilul 474b/1 memorează cererea de schimbare tact CP sincron cu CLK ;

— bistabilul 474b/13 (respectiv 474a/13) este validat (sau blocat) prin intrarea R ;

— sincron cu tactul Tx (12 MHz) respectiv x/1 (7 MHz) acest bistabil comută validînd bistabilul 474c, sau 474c/13 și în același timp suspendînd tactul CLK existent (în "1") ;

— următorul front crescător al lui Tx respectiv x/1 corespunzător CP va relua generarea CLK pe noua frecvență.

Tactul se mai comută pe 3,5 KHz pe durata fiecărei operații de I/O pentru a putea folosi circuite de I/O programabile lente.

3. LOGICA DE SELECȚIE MEMORIE

Circuitul 74LS138/c decodifică semnalele A14 și A5 cointeresat cu RFSH negat (nu se selectează memoriile în timpul refresh-ului) și IORQ negat (se consideră că atunci cînd nu există cerere de lucru cu perifericele avem cerere de acces la memorie).

Semnalul RAM/0 indică lucrul în primii 16 Ko de memorie — 0000-3FFF iar corespunzător RAM/1 negat indică zona 4000-7FFF.

Pentru selecția EPROM-urilor se folosește circuitul LS138b care decodifică A11, A12, A13 cointeresat cu RAM/0, în condițiile ROM * = "0" — sistemul lucrează EPROM ; ROMD negat = = "1" sistemul lucrează cu EPROM-uri din interiorul carcasei.

Diagrama de semnale corespunzătoare selecțiilor EPROM-urilor este dată în fig. 4.3.

Pentru selecția RAM se folosește semnalul CAS negat. Semnalul RAS negat se generează tot timpul din MREQ negat. Inhibarea semnalului CAS negat se face cînd :

— apare selecție EPROM ;

— MREQ negat = 1 ;

— scriere în primii 16 Ko și PC1, 8255 = 0 ;

— RAMD negat = 0 (dezactivare RAM din exterior.

Diagrama de semnale pentru RAS negat, CAS negat, MUX negat este dată în fig. 2.1.

4. SELECȚIE PORTURI I/O

Porturile I/O sînt selectate de circuitul 74LS138a, din adresele A0, A3, A4 cointeresate cu M1 negat (să nu fie IORQ negat din ciclul de răspuns la întrerupere).

— MREQ considerăm că atunci cînd nu este MREQ negat avem cerere de acces la periferice ;

— IOREQ negat = "1" — nu sînt devalidate circuitele de I/O din sistem.

Acest mod de selecție generează și CS-uri negat false pe perioada cît MREQ negat și IORQ negat sînt inactive, dar acest lucru nu deranjează circuitele programabile deoarece nu apare în acest timp nici RD negat, nici WR negat. Pentru selecția circuitului 74LS174 a fost nevoie de condiționarea și cu IORQ negat din motivul mai sus prezentat.

Portul A din circuitul 8255 se selectează atît pe adresa FE cît și E0.

PB = E2.

PC = E4.

Comandă = E6.

5. LOGICA DE COMANDĂ ROM/RAM

Bistabilul 4 713 este adus la "0" după RESET și sistemul lucrează din EPROM. În continuare conținutul adreselor 0000-3FFF se transferă la adresele 800-BFFF, după care se devalidează ROM (IORQ pe adresa FO = COM ROM) după care se transferă 8000-BFFF→0000-3FFF, se invalidează scrierea în primii 16 Ko cu PC/1, 8255 = 0 și se continuă subrutina de RESET.

Bistabilul 474e este legat ca bistabil T și oricînd resetat.

6. DIALOGUL MICROSISTEM—PROCESOR VIDEO

În cazul în care se face o scriere în RAM pe adresele 4000-5FFF informația se scrie și în RAM sistem și în tampon video (aici și adresa corespunzătoare) și se pune pe "0" sincron cu WR negat bistabilul OP. În continuare procesorul video preia din tampoane informația la momente potrivite pentru el și apoi pune pe "1" bistabilul OP. Dacă apare o nouă cerere de scriere în tamponul video cît timp OP = "1", se va suspenda tactul CLK pînă cînd OP devine "0".

Vezi diagrame fig. 4.2.

7. LOGICA DE ÎNTRERUPERI

Intrarea NMI negat nu este folosită. Pe intrarea INT negat microprocesorul primește întreruperi cu frecvența de 50 Hz în vederea inspectării tastaturii. Aceste întreruperi sînt generate de semnalul y8 (frecvența de 50 Hz) îngustat cu ajutorul bistabilului 474q și IORQ negat.

8. INTERFAȚA DE CASETOFON

În scopul salvării programelor din memoria RAM a calculatorului pe casetă audio, se utilizează un casetofon audio obișnuit. Transferul datelor se poate face în ambele sensuri la o viteză de peste 1 200 biți/s (pînă la 2 400). Schema interfeței pentru casetofon audio (înregistrare-redare) este realizată în jurul unui circuit BM339 de la/la care ajunge informația din 2 biți ai circuitului 8 255. Semnalul generat spre casetofon are un preambul pentru sincronizare, după care urmează semnalul util de informație,

modulat în durată. Structura antetului din capul de fişier este formată din 17 octeţi avînd semnificaţia următoare : primul octet conţine tipul fişierului (program BASIC, program în cod maşină, bloc de date numerice sau alfanumerice) ; următorii 10 octeţi conţin numele de fişier ; următorii 2 octeţi specifică lungimea fişierului ; cei doi octeţi următori dau adresa de început a fişierului ; ultimii 2 octeţi specifică pentru programe BASIC numărul liniei pentru autostart.

La sfîrşitul fişierului, pe ultimii octeţi se înregistrează suma de control a blocului respectiv, verificată automat în cazul citirii fişierului.

V. PREZENTAREA INTERFEŢELOR PENTRU IMPRIMANTE

1. PREZENTAREA INTERFEŢEI PENTRU MIM-40

Forma caracterelor de tipărit se trimite în 5 paşi pe cîte 7 biţi corespunzători punctelor dintr-o coloană a caracterului.

Cronograma semnalelor de dialog se poate urmări în fig. 2.1.1. Semnalul PM cît timp este activ, activează motorul care realizează deplasarea capului de imprimare.

Semnalul CAMA indică calculatorului momentele de conectare a celor două microîntrerupătoare, care marchează starea iniţială şi momentul tipăririi capului de imprimare.

În cronogramă s-a indicat şi zona activă de tipărire a capului, aceasta fiind de 310 ms pentru un rînd. Pe durata zonei active de imprimare, calculatorul trebuie să trimită spre imprimantă datele în impulsuri cu durata de 1.3 mc, din care datele trebuie să fie active 400 microsecunde.

Liniile de interfaţă au următoarele semnificaţii în cazul semnalelor de dialog TIM-S MIM-40. Prin linia de comandă paralelă PB/7 a imprimantei se activează motorul imprimantei (PM). Testarea microîntrerupătorului se face pe linie de "răspuns paralel" PA/5 în vederea marcării momentului din care se pot trimite datele spre imprimantă în cadenţa specificată. Pentru trimiterea datelor se folosesc primii 7 biţi din portul B (PB/6—0).

PROGRAMELE DE COMANDĂ

Tipărirea unui rînd de caractere s-a rezolvat prin 3 subprograme : subprogramul TIP-COL care trimite datele pentru tipărirea unei coloane, ţinînd cont de timpii specificaţi în cronograma 2.2.1. pentru starea activă şi inactivă a datelor ; subprogramul RIND care efectuează tipărirea rîndului.

SUBPROGRAMUL CAMA

Organigrama — fig. 2.3.1.

Această rutină marchează momentul începerii tipăririi prin următorii paşi :

— activează motorul poziţionînd pe linia 1 de comandă (bitul 7 din portul B) ;

— testează comutarea microîntrerupătorului şi elimină oscilaţiile produse de acesta prin numărarea a 32 de stări a semnalului CAMA în starea inactivă ;

— testează recomutarea microîntrerupătorului şi elimină oscilaţiile produse de acesta prin numărarea a 256 de stări a semnalului CAMA în starea activă, momentul în care poate începe tipărirea.

SUBPROGRAMUL TIP-COL

Organigrama — fig. 2.3.2.

Cei 7 biţi de date de transmis din registrul D se trimit spre imprimantă prin cei 7 biţi de la 0 la 6 a portului B din circuitul 8255. Comanda de pornire a motorului rămîne în continuare activă. Se păstrează datele active timp de 400 microsecunde prin bucla de întîrziere determinată de contorul din registrul B. În următorul pas se dezactivează datele pentru un timp de 900 microsecunde, bitul pentru activarea motorului fiind în continuare în starea activă.

În continuare se testează dacă s-a cerut întrerupere de la tastatura calculatorului, prin apelarea subrutinei BREAK KEY, care poziţionează fanionul CARRY pe 1 în caz definitiv. În cazul în care s-a cerut întrerupere, se dezactivează motorul şi se iese din program prin rutina BR-K-ERR. În caz contrar se revine prin RET la programul apelat (RIND).

SUBPROGRAMUL RIND

Organigrama — fig. 2.3.3.

Subprogramul realizează tipărirea unui rînd de 32 de caractere obţinînd biţii de date corespunzători prin copia punct cu punct a matricii de caracter din memoria video sau memoria tampon a calculatorului.

Informatia pentru coloana de tipărit se va obţine în registrul D în următorul fel :

Registrul pereche HL conţine adresa de bază a caracterului din memoria video sau tampon. În funcţie de memoria din care se face copia, adresa următorului rînd din caracter se obţine prin incrementarea registrului H cu 1 (în cazul ecranului) respectiv L cu 32 (în cazul memoriei tampon).

Registrul C se foloseste ca mască, si are poziţionat pe 1 singur bit, corespunzător coloanei care se trimite. În momentul iniţial, registrul C are bitul pe 1 şi informatia primului rînd din caracter se află în acumulator.

Registrul B indică numărul coloanei (5 pentru matricea caracterului si 2 pentru spaţiul dintre caractere). Printr-un SI al acumulatorului cu masca din registrul C se testează starea bitului corespunzător si se poziţionează conform acestuia bitul 6 din registrul D. Se aplică aceeaşi mască şi următorilor 6 octeţi (6 rînduri/caracter) obtinuţi prin incrementare corespunzătoare a registrului pereche BC şi se poziţionează bitul 6 din registrul D corespunzător, după rotirea registrului la dreapta. După 7 rotaţii, registrul D conţine informaţia unei coloane din caracter, care se trimite la tipărire prin subprogramul TIP-COL.

2. INTERFAȚA PARALELĂ PENTRU IMPRIMANTE SCAMP

Interfața paralelă este compatibilă CENTRONIX și poate accepta caractere ASCII în paralel pe 7 sau 8 biți.

Semnalele de dialog se pot urmări pe fig. 3.2. Datele stabilizate pe liniile de date trebuie însoțite cu un semnal de strob "DATA STROBE". Cît timp strobul este activ imprimanta își introduce datele într-un registru de reținere și emite semnalul de achitare ACKNOWLEDGE. Cu 5 microsecunde după dezactivarea semnalului BUSY, imprimanta emite semnalul de achitare. ACKNOWLEDGE. Utilizatorul deci poate testa sfîrșitul recepționării de date atît testînd semnalul BUSY cît și ACKNOWLENDE după preferință. Polaritatea celor 3 semnale se poate alege de la microcomutatoarele 8, 9, 10 din grupul C (SCAMP-B-3). S-a ales pentru TIM-S polaritatea din fig. 3.2.

CONDIȚIONĂRI DE TIMP

1. Datele trebuie să fie stabile cu minim 500 ns înainte de activarea semnalului DATA STROBE și încă minim 500 ns după dezactivarea semnalului DATA STROBE. DATA STROBE la rîndul său trebuie să dureze minim 500 ns.

2. BUSY se activează cu maxim 200 ns după DATA STROBE ; devine activ și durează aproximativ 500 ns dacă memoria tampon nu este plină. În cazul în care memoria tampon este plină, BUSY rămîne activ pînă la golirea acestuia.

Semnalele de dialog sînt de nivel TTL.

Corespondența pinilor la cei 36 pini ai cuplei imprimantei se pot urmări în tabelul de mai jos precum și corespondența pinilor la cupla de la calculator.

SEMNALUL	CUPLA IMPRIMANTĂ PIN SEMNAL	CUPLA IMPRIMANTĂ PIN MASĂ	CUPLA CALCULATOR PIN SEMNAL	CUPLA CALCULATOR PIN MASĂ
DATA 1	2	20	1	14
DATA 2	3	21	2	15
DATA 3	4	22	3	16
DATA 4	5	23	4	17
DATA 5	6	24	5	18
DATA 6	7	25	6	19
DATE 7	8	26	7	20
DATE STROBE	1	19	8	21
BUSY	11	29	9	22

3. INTERFAȚA SERIE RS-232-C

S-a ales protocolul de transmisie prin testarea semnalului DATA TERMINAL READY (DTR), acest mod de transmisie fiind mai des utilizat și la alte imprimante. Imprimanta primește date cît timp semnalul DTR este activ. În momentul în care bufferul imprimantei este plin, acest semnal se dezactivează indicînd că este ocupat și nu mai poate primi date de la calculator.

Nivelele de tensiune pentru "0" respectiv "1" sînt +12 V respectiv —12 V. Pentru obținerea acestor tensiuni din nivelele TTL s-au folosit convertoarele ROB 1489 respectiv ROB 1488.

Corespondența pinilor la cei 25 de pini ai cuplei imprimantei se pot urmări în tabelul de mai jos (pinii nefolosiți nu se specifică), alături de corespondența cu pinii cuplei de la calculator.

SEMNALUL	CUPLA IMPRIMANTĂ PIN	CUPLA CALCULATOR PIN
RECEIVER DATA	3	2
FRAME GROUND	1	7
SIGNAL GROUND	7	7
DATA TERMINAL READY	20	20

PROGRAMELE DE COMANDĂ

Imprimata SCAMP poate lucra in 3 regimuri : paralel, serie și grafic, ceea ce presupune 3 seturi de programe de comandă.

PROGRAMUL DE COMANDĂ PENTRU REGIM SERIE — SERIE

Acest program este comun pentru toate imprimantele seriale.

PROGRAM DE COMANDĂ PENTRU REGIM PARALEL TIP — SCAMP

Organigrama — fig. 3.3.2.

Acest program se utilizează pentru trimiterea informației din acumulator spre ieșirile paralele ale portului B din 8255. În cazul transmisiei în regim paralel această informație reprezintă codul ASCII al caracterelor, complementat în rutina P-SCAMP. Se salvează informația în registrul C și se testează prin BREAK-KEY dacă s-a cerut întrerupere de la tastatură. În caz afirmativ se iese din program prin rutina de eroare, în caz contrar se testează semnalul BUSY. Dacă BUSY este inactiv se trimit datele spre portul de ieșire B în 3 pași : cu strob inactiv, cu strob activ, cu strob inactiv.

PROGRAMELE DE COMANDĂ PENTRU REGIM GRAFIC

În cazul acestui regim, datele se cer pe cîte 6 biți, reprezentînd cîte o coloană de 6 puncte din imagine. Ca mod de transmitere a informației grafice s-a ales transmisia paralelă. Pentru a intra în acest mod de lucru, imprimanta care trebuie să primească un cod de comandă ASCII (ESC G) înainte de începerea tipăririi. Pentru trimiterea acestui cod se apelează subrutina COD-GR-SCAMP. Informația grafică se obține din memoria buffer prin copia și transmiterea informației pe coloane de cîte 6 puncte cu ajutorul programului RIND-GR-SCAMP. Pentru trimiterea informației spre imprimantă se apelează rutina tip TIP-SCAMP.

SUBPROGRAMUL COD-GR-SCAMP

Organigrama — fig. 3.3.3.1.

În această subrutină se trimit cei 3 octeţi de comandă de intrare a imprimantei în mod grafic spre aceasta (ESC G, în ASCII 1B47).

SUBPROGRAMUL RIND-GR-SCAMP

Organigrama — fig. 3.3.3.2.

Cu ajutorul acestui subprogram se obţine în registrul C informaţia corespunzătoare a cîte 6 biţi, informaţia (osteţi organizaţi în linie şi nu în coloană) obţinîndu-se din memoria tampon de la adresa 5B00. Biţii din care se formează coloana se testează pe bitul 7 al fiecărui octet, informaţia rotindu-se la stînga pînă trec toţi biţii din octat prin poziţia 7. Cînd începe testarea unei coloane de 6 octeţi, se poziţionează al 6-lea bit din registrul C pe 1. Valoarea bitului 7 din primul octet se poziţionează pe bitul 7 din registrul C, următorul tot pe bitul 7, dar după o deplasare prealabilă la dreapta a registrului c. După 6 deplasări apare depăşire şi se apelează subprogramul de tipărire. Se continuă cu următorul caracter pînă cînd se parcurg toţi cei 256 octeţi după care se trimite codul de sfîrşit de linie.

INTERFAŢA ŞI PROGRAMELE DE COMANDĂ PENTRU IMPRIMANTA DZM 180.

Transmiterea caracterelor spre imprimantă se face în cod ASCII, pe 7 biţi, fără control de paritate. Semnalele de dialog dintre TIM-S şi imprimantă, cît şi corespondenţa lor la cuplă se dă în fig. 4.2.1. Din aceste semnale sau folosit următoarele :

— 7 linii de dat : ENT1-ENT7 negaţi ;

— 1 linie de validare a datelor : SE negat ;

— 1 linie de răspuns de prelucrare a datelor : ACK negat.

Cronograma acestor semnale se dă în fig. 4.2.2. Transmiterea datelor spre imprimantă se face activînd simultan şi linia SE, iar după ce imprimanta a răspuns prin ACK, preluare date, se dezactivează, SE, iar datele nu mai sînt necesare.

Toate semnalele sînt de tip TTL.

Liniile de date sînt conectate la ieşirile portului B inversoare cu rol de amplificatoare de linie. Linia SE este conectată la bitul 7 al portului B iar linia de răspuns ACK la bitul 7 al portului A.

PROGRAMUL DE COMANDĂ TIP-V&DZM

Organigrama — fig. 4.3.

Programul de comandă TIP-V&DZM are rolul de a asigura generarea semnalelor necesare pentru transmiterea unui caracter de la TIM-S la imprimanta DZM. Caracterul ce trebuie transmis se află în registrul B reprezentat în cod ASCII pe 7 biţi şi trebuie trimis la portul de ieşire D, împreună cu un semnal de validare a datelor (SE negat). Se aşteaptă apoi răspunsul imprimantei (ACK negat), după recepţionarea semnalul de comandă. De fapt se anulează tot octetul înscris în port înscriindu-se 0. Se impune o salvare a registrelor care se folosesc în această subrutină.

Deoarece setul de caractere al imprimantei nu cuprinde literele mici, codurile acestora sînt convertite în codul literei mari corespunzătoare pentru a permite tipărirea oricărui text. Acesta se realizează prin resetarea bitului 5 din acumulator, ceea ce echivalează cu o scădere de 20/H din acumulator.

După ce imprimanta a preluat caracterul, şi a răspuns prin ACK, se reîncarcă registrele din stivă şi se face o întoarcere în programul apelant.

Complementarea datelor nu este necesară deoarece inversarea liniilor este asigurată de inversoarele CDB 404 de pe liniile de date.

PREZENTAREA INTERFEŢEI ŞI PROGRAMELE DE COMANDĂ PENTRU IMPRIMANTA VIDEOTON

Imprimanta ES 7184 produsă de firma VIDEOTON (R.P.U.) este o imprimantă cu tambur, cu performanţe ridicate (800 linii/minut).

Transmiterea caracterelor spre imprimantă se face în cod ASCII, pe 7 biţi, fără control de paritate. Semnalele de dialog între imprimată şi TIM-S sînt :

— 7 linii de date : DATA1-DATA7 ;

— 1 linii pentru strobarea datelor : DATA STROBE ;

— 1 linie de răspuns : READY.

Forma acestor semnale se arată în cronograma din fig. 5.2.1. a şi cronograma de dialog în fig. 5.2.1. b.

Corespondenţa semnalelor la cuplă se dă în fig. 5.1.2 (tab. 5.1.2).

Nivelele logice cerute de imprimantă sînt :

— 0 V pentru "0" ;

— +5 V pentru "1".

Liniile de date sînt conectate la portul B prin circuite inversoare pentru amplificarea semnalului. Linia RUF corespunde bitului PB67 şi linia de răspuns bitului PA7.

PROGRAMUL DE COMANDĂ TIP-V&DZM

Deoarece imprimanta VIDEOTON are acelaşi tip de dialog cu TIM-S ca şi imprimanta DZM180, se foloseşte programul de comandă de la această imprimantă.

INTERFAŢA ŞI PROGRAMELE DE COMANDĂ PENTRU CONSOLA "CENTRONIX"

Transmiterea caracterelor se face în serie, cu o rată de transmisie de 300 baud. Imprimanta este prevăzută cu o interfaţă RS 232, avînd nivelele de tensiune :

"0" — pentru +3—+12 V ;

"1" — pentru —3——12 V.

Linia de date este legată la PC/0 prin intermediul unui convertor TTL+—12 V de tip ROB 1488, care asigură nivelele de tensiune cerute de standard-ul RS 232.

Linia de răspuns a imprimantei, BUSY este legată printr-un convertor, +—12 V TTL de tip ROB 1489 la PA/5.

4. PROGRAMUL DE COMANDĂ SERIE

Organigrama — fig. 6.3.

Subrutina SERIE realizează serializarea biților din octetul cod ASCII conținut în registrul B. Se testează semnalul READY obținut prin bitul 5 al portului A din circuitul 8255 și se rămîne în așteptare dacă este inactiv.

În caz contrar se dezactivează întreruperile și se trece informația în registrul acumulator. Pentru bitul start se introduce 0 pe cel mai puțin semnificativ bit al acumulatorului prin rotire la stînga. Se trimite bitul start spre portul C al interfeței și se apelează rutina de întîrziere TIPM care caracterizează viteza de transmisie. În continuare se trimit cei 7 biți de date prin rotirea acumulatorului spre dreapta. După 7 biți de date se poziționează cel mai puțin semnificativ bit de acumulator pe 1 și se transmit cei 12 biți pe STOP. După ce se reface conținutul lui B se activează întreruperile și prin RET se revine la programul apelant.

SUBPROGRAM TIMP

Organigrama — fig. 6.3.1.

Este o rutină de întîrziere prin decrementarea registrului B. Constanta din registrul B determină viteza de transmisie.

PREZENTAREA INTERFEȚEI PENTRU IMPRIMANTA ROBOTRON K 6311

Semnalele de dialog și semnificația lor se poate urmări în fig. 7.2.1 iar cronograma semnalelor în fig. 7.2.2.

Protocolul folosit este identic cu cel de la imprimanta SCAMP putîndu-se folosi atît interfața paralelă cu protocolul tip "BUSY" cît și programul de comandă al acestuia, cu diferența că datele nu trebuie complementate ca la imprimanta SCAMP.

Interfața serie. Datele sînt transmise la portul PC/0 iar dacă tamponul imprimantei este plin, acesta răspunde prin dezactivarea liniei DTR la portul A bitul 5.

Semnalele de interfață sînt reprezentate în fig. 7.2.3, iar cronograma lor în fig. 7.2.4.

PROGRAMUL DE COMANDĂ SERIE

Pentru comanda acestei imprimante se folosește o rutină comună pentru toate imprimantele serie, rutină numită SERIE tratată la capitolul respectiv. Organigrama se poate urmări în fig. 6.3.

VI. SISTEMUL DE OPERARE

1. ÎNLĂNȚUIREA PROGRAMELOR

Pentru deschiderea unui canal de la 4 la 15 se face uz de un tabel DATA STREAM care în octeții corespunzători numărului de canal păstrează un deplasament necesar obținerii unor informații din tabelol CHANNEL INFORMATION asupra canalului respectiv.

Deschiderea unui canal se realizează apelînd la subrutina OPEN N, "x". Dacă N este un număr mai mic decît 15 și codul canalului x se găsește în tabelele OPEN STREAM LOOK UP (adică există acest canal) rutina introduce în cei 2 octeți corespunzători numărul canalului din DATA STREAM deplasamentul pentru informația de canal.

La comanda PRINT N, se intră în rutinele de tratare a comenzii PRINT. Dacă se găsește numărul de canal specificat, se apelează rutina CHAN-OPEN, care verifică dacă este deschis canalul N (dacă are octeții de deplasament în DATA STREAM). Pe baza acestui deplasament se calculează adresa informației de canal (tab. CHANNEL INFORMATION), corespunzător canalului x, și se memorează la variabila de sistem CURCH. Parcurgînd în continuare rutinele din RST 10, pe baza adresei păstrate în CURCH, se obțin adresele programelor pentru tipărire din memoria tampon sau memoria video.

a. În cazul copiei de ecran (cod canal majusculă) se face saltul direct la rutinele specifice fiecărei imprimante de copie de memorie video. Aceste rutine sînt distincte pentru SCAMP grafic ("G" — adresă 3975-ECRAN-GR-SCAMP) pentru MIM-40 ("M" — adresă OEAC-COPY-ECRAN-MIM-40), și comună pentru celelalte canale ("A", "B", "X"-3AF4-ECRAN P-S).

b. În cazul copiei memoriei tampon (cod canal literă mică) canalele sînt tratate în continuare împreună, adresa comună fiind 09F4, parcurgînd rutinele necesare obținerii informației în buffer, și ramificarea se face cînd se ajunge la adresa rutinei COPY BUFFER pentru imprimanta ZX-PRINTER printr-un salt la adresa 3B6A, la subprogramul ID-g-m. Dacă s-a deschis canalul g respectiv m, acestea avînd o tratare specială, se apelează programele lor de copie tampon ("g" — adresă 3921-BUFF-GR SCAMP, "m"-adresă OEDO-COPY-BUFFER-MIM-40). Pentru celelalte canale se apelează o rutină comună de copie buffer ("a", "b", "x"-adresă 3ADO-BUFF-CH-P-S).

Identificarea canalelor se efectuează prin apelarea programului de identificare canal IDCH din rutinele BUFF-CH-P-S respectiv ECRAN-P-S.

În cazul comenzii COPY s-a modificat adresa de intrare în rutina COPY a lui SCAMP (ECRAN-GR-SCAMP-"G") tipărirea obținîndu-se la imprimanta SCAMP în mod grafic.

LLIST și LPRINT comunică pe canalul standard P pentru care în rutina IDCH s-a indicat adresa de salt corespunzător canalului "b", adică se transmit datele la SCAMP paralel prin copie buffer. Înlănțuirea programelor se poate urmări în fig. 8A.2.

2. PROGRAMELE PENTRU TRATAREA OPERAŢIILOR DE I/O

2.1. Tabelele utilizate

NOTĂ : Pentru a evita confuziile posibile dintr-o traducere incorectă, s-au păstrat denumirile originale din limba engleză.

La iniţializarea sistemului unele din tabelele (cele numite iniţiale) se transferă din memoria EPROM în RAM pentru a permite unele modificări eventuale în cursul programării. Astfel tabelul INITIAL STREAM DATA se transefă de la adresa 15C6 la 5C10 pe lungimea de 14 octeţi, iar tabelul INITIAL CHANNEL INFORMATION se transferă de la adresa 15AF la 5CB6 pe lungime de 14 octeţi.

Tabelul STREAM DATA este organizat pe cîte doi octeţi pe o lungime de 14+2×12 începînd de la adresa 5C10. Adresele perechilor de octeţi se succed din 2 în 2 începînd cu adresa 5C16. Cei 2 octeţi conţin un deplasament care adăugat la constanta 5C16 de la adresa dată de variabila de sistem CHANS (5C4F, 5C50) dă adresa informaţiei de canal CHANNEL INFORMATION. Tabelul CHANNEL INFORMATION este organizat pe cîte 5 octeţi şi se află în zona de memorie 5CBC la 5CCA pentru canalele "K", "S", "P" şi 3A40 la 3A71 pentru canalele de imprimante. Primii 4 octeţi din cei 5 sînt adresele rutinelor de tipărire pentru transfer date de tip emisie respectiv recepţie, iar ultimul octet este codul canalului.

Tabela OPEN STREAM LOOK UP este organizată pe cîte 2 octeţi pentru fiecare canal plus un octet de marcaj la sfîrşitul tabelei pe 0. Pentru canalele "K", "S", "P" această tabelă se găseşte între adresele 177A şi 1780, iar pentru canalele de tipărire la imprimante între adresele 3A00 şi 3A14. Primul din octeţi este codul canalului, iar al doilea este un deplasament, care adunat la adresa octeţilor duce la rutinele OPEN"X".

Tabela CHANNEL CODE LOOK UP situat în zona de memorie 162D la 1633 este organizată identic cu tabela OPEN STREAM LOOK UP prin deplasamente de subrutinele CHANNEL "X"FLAG.

2.2. Rutina OPEN # X'

Organigrama — fig. 8A.3.2.

Prin intermediul lui RST 28 şi a subprogramului STR-DATA se obţine în registrul pereche din tabelul DATA STREAM conţinutul celor 2 octeţi de deplasament corespunzători numărului de canal N. Verifică dacă acestea sînt pe 0 sau nu respectiv dacă canalul este închis sau deschis. Dacă găseşte canalul deschis (poate fi deschis şi ataşat unui alt nume) se mai parcurg următorii paşi înainte de deschidere. Se calculează în HL adresa de intrare în tabelul CHANNEL INFORMATION prin adunarea deplasamentului din DATA STREAM la adresa de bază 5CB6 obţinut de la adresa CHANS (5C4F, 5C50) şi se ia de la adresa dată în HL incrementat cu 4 codul canalului vechi. Se verifică dacă acesta coincide cu unul din codurile posibile, şi în caz contrar se afişează eroarea. Pentru a verifica pe lîngă canalele standard "K", "S", "P", şi cele nou introduse, s-a impus o modificare care prin apelul subprogramului IDENT-CH (şi prin acesta a subprogramului COR-OPEN-CD) caută codul canalului vechi în tabelele OPEN STREAM LOOK UP de la adresele 177A respectiv 3A00.

În cazul în care nu se găseşte se afişează eroare, iar în cazul în care este corect, se trece la redeschiderea canalului, pentru codul de canal specificat în instrucţiune. Se apelează subprogramul OPEN-2 în care se calculează noul deplasament pentru DATA STREAM. Înainte de întoarcerea la programul din care s-a apelat se încarcă deplasamentul obţinut în registrele DE în tabelul STREAM DATA.

2.2.1. *Rutina OPEN-2*

Organigrama — fig. 8A.3.2.1.

La intrarea în această rutină registrul pereche HL conţine adresa octeţilor din STREAM DATA iar B şi C octeţii de deplasament. După salvarea conţinutului în HL, prin intermediul subprogramului FETCH se extrage lungimea numelui canalului în registrul pereche BC iar adresa primului caracter din nume în DE. Se generează un raport de eroare pentru nume canal de lungime zero. Se apelează COR-OPEN-CD cu adresa de bază pentru tabelul OPEN STREAM LOOK UP (177A) care va căuta în cele 2 tabele de la 177A respectiv 3A00 codul canalului nou. Dacă nu se găseşte, înseamnă că s-a cerut deschiderea unui canal inexistent şi se generează un raport de eroare. Dacă s-a găsit codul, HL va indica adresa deplasamentului corespunzător la care se adună valoarea deplasamentului. Saltul la conţinutul lui HL duce la rutinele OPEN 'X' corespunzător fiecărui canal iar de aici la rutinele OPEN END de la adresa 18B respectiv 3A70. Aceste rutine generează un raport de eroare dacă găsesc că lungimea numelui de canal conţine mai mult de un caracter şi încarcă în D partea mai semnificativă a octetului de deplasament din DATA STREAM (00 pentru canalele K, S, P şi DD pentru canalele de imprimante).

2.2.2. *Rutinele IDENT-CH şi COR-OPEN-CD*

Organigrama — fig. 8A.3.2.2a, 8A.3.2.2b.

În subprogramul IDENT-CH se încarcă partea mai puţin semnificativă a adresei de bază pentru tabelul OPEN STREAM LOOK UP (177A). Subprogramul COR-OPEN-CD apelează subrutina INDEXER şi caută codul canalului dat în registrul C în cele 2 tabele cu adresele de bază 177A respectiv 3A00.

2.2.3. *Rutina STR-DATA*

Organigrama — fig. 8A.3.2.3.

Acest subprogram obţine numărul canalului (N) din stiva calculatorului prin intermediul subprogramului STK-TO-A în acumulator. Ve-

rifică dacă acest număr nu a depășit 16 și în caz afirmativ se generează eroare. Dacă numărul este corect, pe baza lui N calculează adresa de intrare în tabelul STREAM DATA în registrul pereche HL și încarcă deplasamentul în registrul BC. Se revine prin RET.

3. RUTINA CLOSE # N

Organigrama — fig. 8A.3.3.

Această rutină obține în primul pas prin intermediul subrutinei STR-DATA cei 2 octeți de deplasament din STREAM DATA în BC și adresa acestora în HL. Apelează CLOSE-2 pentru a verifica dacă există în tabele CLOSE STREAM LOOK UP (adresa bază 171A) respectiv OPEN STREAM LOOK UP (adresa bază 3A00) codul corespunzător și în caz contrar generează un mesaj de eroare. La adresa octeților din DATA STREAM se adună A3E2. Dacă în urma acestei adunări apare depășire înseamnă că numărul canalului este mai mare decît 3, deci urmează să se deschidă acel canal prin înscrierea la adresa din STREAM DATA a valorii 00 din registrul pereche BC. Dacă CARRY rămîne pe 0, înseamnă că este un canal standard care nu poate fi închis și urmează să se încarce informația din tabelul INITIAL DATA STREAM a cărei adresă se obține prin adunarea constantelor A3E2 și 15D4 la adresa octeților din DATA STREAM.

SUBPROGRAMUL CLOSE-2

Organigrama — fig. 8A3.3.1.

La intrarea în această rutină BC conține cei 2 octeți din STREAM DATA pentru canalul numărului N iar HL adresa acestuia. Se formează în HL (după salvare) adresa informației de canal prin adunarea adresei de bază (5CA6) obținută de la adresa CHANS (4F2A, 4F2B) cu deplasamentul din DATA STREAM. Se incrementează adresa HL pînă cînd se ajunge la adresa codului de canal ce se încarcă în C. După salvarea adresei codului de canal, se caută prin intermediul subprogramului INDEXER codul canalului în tabelul CLOSE STREAM LOOK UP și cu ajutorul deplasamentului se intră în subrutina CLOSE STREAM care reîncarcă HL din stivă și se revine prin RET.

Pentru a parcurge și tabelul OPEN STREAM LOOK UP pentru canalele de imprimante se apelează în loc de subrutina INDEXER rutina COR-OPEN-CD prin intermediul rutinei COR-CLOSE. Dacă nu se găsește cod canal cunoscut se generează un raport de eroare. Rutina se termină prin reîncărcarea în HL a adresei octeților STREAM DATA.

4. RUTINA CHAN-OPEN

Organigrama — fig. 8A.3.4.

La intrarea în această subrutină registrul acumulator A conține numărul canalului (#N). Pe baza acestui număr și a adresei de bază a tabelului STREAM DATA se obține în HL adresa celor 2 octeți de deplasament (offset) din tabelul STREAM DATA care caracterizează acest canal. Cei 2 octeți de deplasament se încarcă în registrul DE. Dacă ambii octeți au valoarea zero înseamnă că acest canal nu a fost deschis și se generează un mesaj de eroare.

Deplasamentul din tabelul STREAM DATA adunat cu adresa de bază pentru informația de canal (5CB6), obținută de la adresa dată de variabila de sistem CHANS (5C4F), formează adresa de intrare în tabelul CHANNEL INFORMATION, la zona de tabel corespunzător codului de canal (x) pentru care s-a deschis canalul N. Această adresă a canalului curent se păstrează la adresa dată de variabila de sistem CURCHL (5C51). Se incrementează adresa din tabelul cu informația de canal pînă se ajunge la codul canalului, care se încarcă în registrul C. În continuare se încarcă adresa de intrare pentru tabelul CHANNEL COD LOOK UP TABLE (1621) unde se caută codul canalului dat în C pentru a obține un deplasament necesar în continuare. Pentru găsirea codului se face apel la subprogramul INDEXER, care poziționează CARRY pe 1 dacă găsește codul canalului și în HL dă adresa deplasamentului corespunzător ; se poziționează CARRY pe 0 în cazul în care nu găsește codul (ajunge la marker=00). Pe baza acestui offset și adresa acestuia se ajunge pentru fiecare canal în parte la cîte o subrutină numită CHANNEL FLAG printr-un CALL la adresa calculată. În cazul că s-a ajuns la marker, în tabelul CHANNEL CODE LOOK UP TABLE, înseamnă că s-a deschis un canal în afara celor standard.

Pentru apelul subrutinei CHANNEL FLAG în acest caz, s-a modificat parametrul de apelare a subprogramului INDEXER în subprogramul COR-CHAN-OPEN (3A9D).

Organigrama — fig. 8A.3.4.1.

La intrarea registrului pereche HL conține adresa bază pentru tabel iar registrul C codul canalului. Informația din tabel fiind organizată pe cîte 2 octeți : offset și cod canal, programul caută prin incrementarea adresei codul canalului din registru C. Dacă nu găsește și ajunge la marker (00) se revine cu RET în programul apelant cu fanionul CARRY pe 0. Dacă codul este identificat se poziționează CARRY pe 1 și se revine prin RET, cu adresa offset-ului în HL.

4.2. **COR-OPEN-CHAN**

Organigrama — fig. 8A.3.4.2.

La intrare registrul HL conține adresa de bază a tabelului iar acumulatorul codul canalului. Se apelează subprogramul INDEXER. Dacă la revenire CARRY este 0 (adică nu s-a găsit în tabel codul canalului căutat) înseamnă că în registrul C se găsește codul unui canal de tipărire la imprimantă, și înainte de revenirea în program setează bitul 1 din fanionul FLAGS, semnalînd prin aceasta că s-a deschis un canal pentru tipărire la imprimantă.

5. RUTINELE PRIN, LPRINT, (PRINT #N, LPRINT #N)

Înlănţuirea programelor necesare tipăririi prin comenzile PRINT, LPRINT se poate urmări pas cu pas pornind de la adresele 1FCD respectiv 1FC9. Prezentarea acestor rutine se va rezuma la strictul necesar pentru înţelegerea diferenţei dintre acestea şi cele cu # N.

La intrare în această rutină se pregăteşte în acumulator numărul canalului pentru S(01) şi P(03) funcţie de tipărire ecran (PRINT) sau imprimantă (LPRINT). Prin rutina OPEN-CHANNEL se actualizează conţinutul adresei CURCHL cu adresa informaţiei de canal actual (N=2 sau 3) din CHANNEL DATA. După parcurgerea mai multor rutine se ajunge la un apel al subrutinei STR-ALTER, care dacă găseşte că în comandă s-a specificat numărul de canal, pe baza acestuia actualizează din nou CURCH cu adresa informaţiei de canal corespunzător acestui canal. După parcurgerea altor rutine se apelează, cu octetul de tipărit în acumulator, RST 10.

La apelarea acestei rutine se încarcă în registrul pereche HL de la locaţia CURCHL (canal curent) adresa canalului curent (din tabelul CHANNEL INFORMATION cu adresele de bază 5CBC respectiv 3A40) şi se face un salt indirect la 09F4. În cazul canalelor pentru copie memorie tampon (buffer) caracterizate prin nume caracter mic, se apelează aceeaşi subrutină de la 09F4. Ramificarea pentru diferenţele imprimante s-a făcut prin introducerea unui salt la 3B6A (rutina I-g-m). În cazul în care se cere copie de ecran (cod canal majusculă) ramificarea la diferitele programe de tipărire se face deja direct din tabelul cu informaţia de canal.

6. RUTINELE LIST, LLIST (LIST #N, LLIST #N)

Înlănţuirea acestor programe se poate urmări de la adresele 17F9 respectiv 17F5. Aceste comenzi apelează la final rutinele PRINT, LPRINT.

7. SUBRUTINA SCREEN $

Organigrama — fig. 8A.3.8.

Această subrutină se apelează indicînd coordonatele unui caracter de pe ecranul TV, X=0,32 şi Y=0,23. La ieşirea din subrutină se obţine în perechea de registre DE adresa locaţiei care conţine codul ASCII corespunzător caracterului ale cărui coordonate s-au dat.

La adresa CHARS+256 se găseşte o tabelă cu matricele caracterelor începînd cu spaţiul (cod ASCII #20) pînă la caracterul căruia îi corespunde codul ASCII 7F. Adresa de început a acestui tabel se încarcă în registrul pereche HL. Adresa primului octet al caracterului în cauză din memoria video se calculează şi se introduce în perechea de registre DE. Astfel, în E se introduce 32 (x mod 8) +y, iar în D 64+8INT (X/8).

Se compară primul octet al caracterului cu primul octet din tabel (corespunzător primei matrice din tabel) atît pentru caracter afişat direct, cît şi pentru cel afişat invers pe ecran. În caz de coincidenţă se compară şi următorii 7 octeţi. La prima necoincidenţă se face salt la următoarea matrice de caracter şi se începe compararea de la primul octet. Registrul B va indica numărul de ordine al unei matrici din cele 96 existente.

Codul primei matrice fiind 20 H iar numărul total de caractere din tabel fiind 60 H=96 Z, 20 H+60 H=80 H, codul corespunzător matricei la care s-a găsit coincidenţa, se calculează prin scăderea din 80 H a conţinutului registrului B.

Modificarea s-a impus pentru introducerea unei secvenţe de program, care să calculeze adresa următorului caracter de tipărit, funcţie de memoria din care se copiază, buffer sau ecran (COR-SCREENS). Subprogramul SCREEN $ iniţial presupunea că se lucrează numai în memoria ecran.

8. SUBRUTINA BR-K-ERR

Se apelează, în cazul în care s-a testat întrerupere de la tasta BREAK. Apelează rutina CLEAR-PR-BUFF pentru a şterge conţinutul memoriei tampon, după care iese din program prin RAPORT pe ecran.

9. RUTINELE PENTRU COPIE BUFFER

9.1. Subrutina BUFF-CH-P-S

Organigrama — fig. 8A.3.10.1.

Acest subprogram trimite informaţia din memoria tampon (zona de memorie 5B00-5BFF) spre imprimante. În acest scop se încarcă adresa de bază a bufferului în registrul pereche DE şi adresa de bază a tabelului cu matricele caracterului în registrul pereche HL. Pe baza adresei matricei caracterului din buffer în DE şi adresei primului caracter din zona de caractere în HL, programul SCREEN $ obţine de la adresa dată de registrul pereche DE codul caracterului în ASCII. Se trimite codul caracterului prin rutina IDCH (identificare canal) şi se generează adresa următorului caracter din buffer. Dacă s-au tipărit deja 32 caractere (20 H) se trimite codul de sfîrşit de linie prin rutina LFCR, şi se şterge bufferul, apelînd la subprogramul CLEAR-PRINTER BUFFER.

9.2. COPY-BUFF-MIM 40

Organigrama — fig. 8A.3.10.2.

Subprogramul efectuează copia memoriei tampon la imprimanta MIM 40. Adresa de bază a buffer-ului 5B00 se încarcă în registrul pereche HL. Transferul de date se efectuează prin subprogramul RIND, iar după efectuarea tipăririi se şterge bufferul apelînd BUFF-CLEAR.

9.3. Rutina BUFF-GRA-SCAMP

Organigrama — fig. 8A.3.10.3.

Această rutină este apelată cînd se doreşte copia memoriei tampon (5B00-5BFF) în mod grafic la imprimanta SCAMP. Prin apelarea rutinei COD GRAFIC, trimite spre imprimantă cei 2 octeţi ASCII ai codului de intrare în mod grafic (ESC G). Se apelează RIND-GR-SCAMP pentru a trimite informaţia la imprimantă pe coloane de cîte 6 puncte. La revenirea din acest subprogram, trebuie tipărite şi ultimele 2 rînduri rămase din matricea de 8 puncte pe coloană. Se transferă zona de informaţie a ultimelor 2 rînduri pe primele 2 rînduri şi se completează celelalte 4 rînduri cu 0, şi se apelează din nou RIND-GR-SCAMP pentru a scrie şi partea inferioară a caracterelor. La sfîrşitul subprogramului se apelează rutina de ştergere a memoriei tampon (CLEAR-BUFFER).

10. RUTINELE PENTRU COPIE DE ECRAN

10.1. Rutina ECRAN-P-S

Organigrama — fig. 8A.3.11.1.

Prin intermediul acestui program se efectuează copia ecranului (memoriei video) la majoritatea imprimantelor (excepţie fiind SCAMP în regim paralel şi grafic şi MIM 40). În registrele B şi C se încarcă coordonatele caracterului X, Y de pe ecran, ecranul fiind considerat ca o matrice de 32 coloane şi 24 rînduri. Pe baza coordonatelor din subprogramul SCREEN$ se obţine codul caracterului la adresa specificată de registrul pereche DE. Cînd se ajunge într-o linie la al 33-lea caracter ((B)= =32), prin subrutina LFCR se trimite codul de sfîrşit de linie, salt la următorul rînd. După trimiterea a 24 de rînduri (ultimul rînd al ecranului) se iese din program prin salt la subrutina CLEAR-BUFFER.

10.2. COPY MIM 40

Organigrama — fig. 8A.3.11.2.

Prin acest subprogram se realizează copia ecranului TV (memoria video) şi trimiterea ei spre imprimanta MIM 40. Adresa de bază (primele locaţii din memorie) 4000 se încarcă în registrul pereche HL. După pornirea imprimantei prin subprogramul CAMA, prin apelul rutinei RIND, se tipăreşte la imprimantă un rînd începînd de la adresa din HL. Pentru contorizarea celor 24 RIND-uri se foloseşte registrul D. După tipărirea ultimului rînd se dezactivează motorul şi se apelează subprogramul CLEAR-BUFF.

10.3. ECRAN-GR-SCAMP

Organigrama — fig. 8A.3.11.3.

Pentru efectuarea copiei de ecran în mod grafic, informaţiile de pe ecran trebuie trimise la imprimantă în cîte 6 rînduri consecutive. Deoarece organizarea adreselor la memoria video este greu de urmărit, (organizarea acestora este pe cîte 8 rînduri şi nu 6) se transferă liniile de date în memoria tampon asamblîndu-se acolo cîte 6 rînduri succesive. Această funcţie revine rutinei COPY-SCAMP. Rutina pregăteşte adresa de bază a ecranului în registrul HL, şi după tipărirea unui rînd de 6 coloane, trimite codul de avans al hîrtiei pentru regim grafic, şi şterge bufferul apelînd subprogramul BUFFER-CLEAR.

10.4.a. TIP-RIND-GRAFIC

Organigrama — fig. 8A.3.11.3.a

La intrarea în această rutină registrul pereche DE conţine o adresă din memoria buffer, în registrul pereche HL adresa din memoria video, iar în registrul B numărul liniilor de transfer. Prin apelul subprogramului TRANS-SC transferă un număr de linii dat de conţinutul registrului B în memoria tampon şi prin apelul subprogramului RIND-GR-SCAMP se tipăresc la imprimantă primele 6 rînduri din memoria buffer. Înainte de întoarcere prin RET la programul apelant (COPY-SCAMP), încarcă în registrul pereche DE adresa de bază pentru memoria buffer (5B00).

10.4.b. SCAD-SC

Organigrama — fig. 8A.3.11.3.b.

Subprogramul efectuează scăderea constantei 07E0 din conţinutul registrului pereche HL pentru obţinerea adresei de bază a primului caracter din următorul rînd de ecran.

10.4.c. COPY-SCAMP

Organigrama — fig. 8A.3.11.3.c.

Această rutină asamblează informaţia grafică pe cîte 6 rînduri în memoria tampon. Informaţia o combină din rîndurile de ecran formate din cîte 8 linii în următorul fel : 6,2 ; 4,4 ; 2,6 ; 6,2 etc. Incrementarea adreselor de memorie se efectuează în programul de transfer a informaţiei din memoria video în memoria buffer (TRANS-SC). În această rutină după transferul ultimului octet se mai incrementează încă o dată registrul HL. Pentru calculul adresei primului octet din următorul rînd de ecran se scade din adresa incrementată HL (cu 100 H mai mult decît adresa ultimului octet transferat) 07E0. La trecerea de la un cadran la altul (L=E0) trebuie refăcută adresa corectă (ex. 48E0 înseamnă de fapt 47E0, ceea ce este adresa ultimului rînd din primul cadran, deci se formează adresa corectă 4800). Tipărirea rîndurilor de cîte 6 linii după asamblarea lor în buffer se efectuează prin apelul rutinei TIP-RIND-GRAFIC.

10.4.d. TRANSF-SC

Organigrama — fig. 8A.3.11.3.d.

La intrarea în această rutină registrul pereche HL conţine adresa zonei de date sursă, registrul pereche DE adresa zonei de date desti-

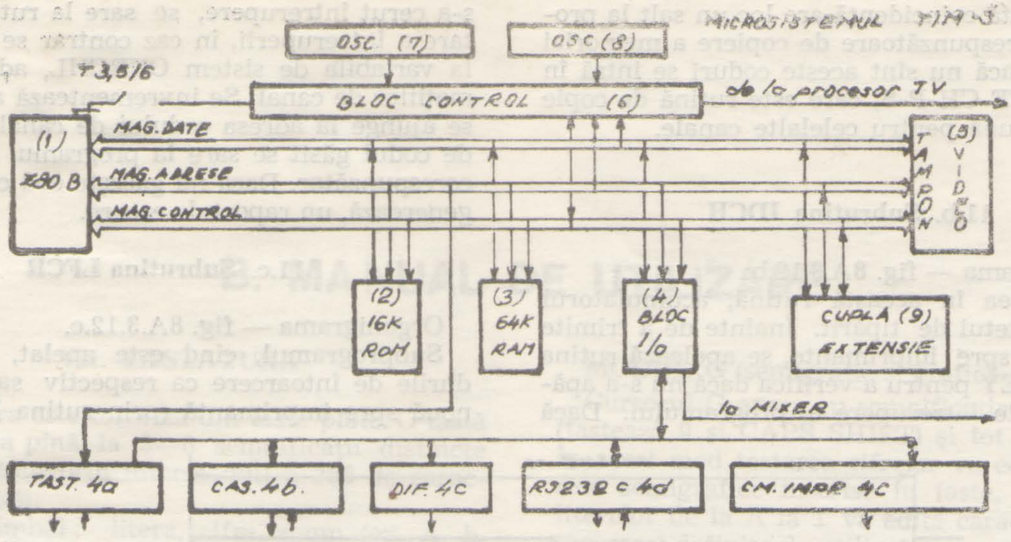


Fig. 1. a

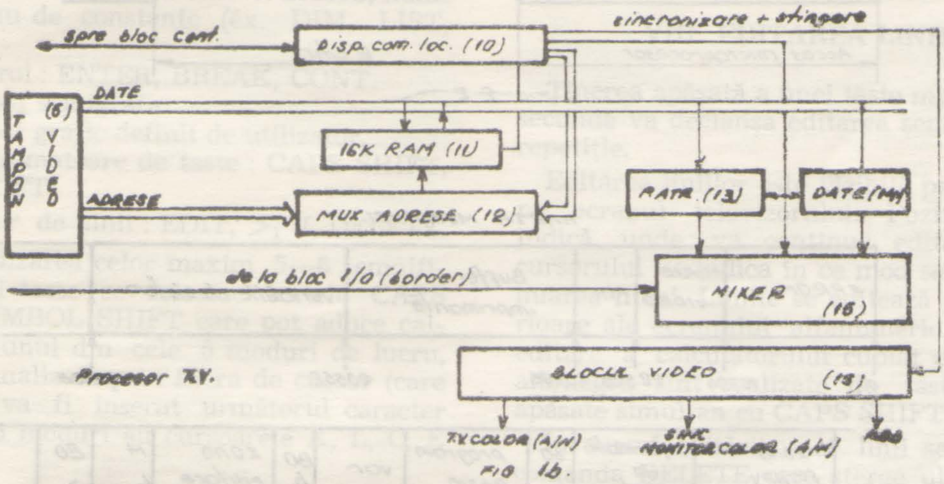


Fig. 1. b

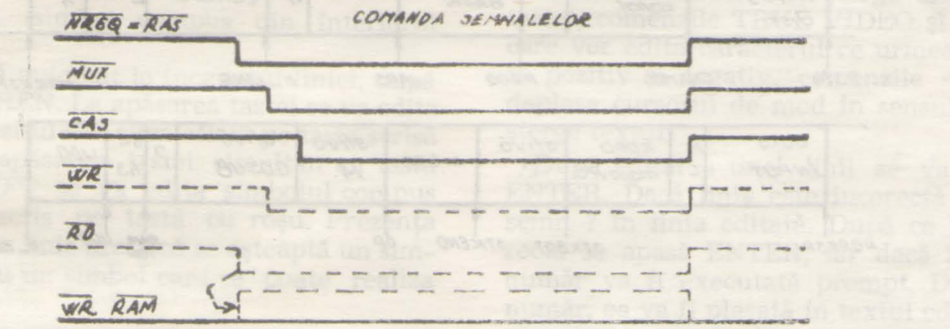


Fig. 2.1

nație, și BC numărul rîndurilor de cîte 32 de octeți de transferat. După terminarea transferului pe cîte 32 de octeți a întregii informații se revine prin RET.

10.5. P-SCAMP

Organigrama — fig. 8A.3.11.4.

Se apelează la transmisia codurilor ASCII spre interfața paralelă. Codul caracterului ASCII primit în registrul B se complementează și după salvarea registrului pereche BC se trimite prin subprogramul TIP-SCAMP la imprimantă. După ce se reface conținutul registrului pereche BC se revine prin RET.

11. RUTINELE PENTRU IDENTIFICAREA CANALULUI

11.a. Subrutina ID-g-m

Organigrama — fig. 3.12.a.

În acest subprogram se intră în cazul în care s-a cerut copie de buffer, pentru a separa tratarea canalelor g respectiv m de celelalte, acestea presupunînd un mod de tratare special. Se încarcă adresa informației de canal curent de la adresa dată de variabile de sistem (CURCHL) în registrul pereche HL. Se incrementează adresa pînă se ajunge la adresa codului de comandă. Codul de comandă din acumulator se compară cu codurile canalelor g respectiv m,

şi dacă există coincidenţă are loc un salt la programele corespunzătoare de copiere a memoriei tampon. Dacă nu sînt aceste coduri se intră în rutina BUFF-CH-P-S, care este rutină de copie buffer comună pentru celelalte canale.

11.b. Subrutina IDCH

Organigrama — fig. 8A.3.12.b.

La intrarea în această rutină, acumulatorul conţine octetul de tipărit. Înainte de a trimite informaţia spre imprimante, se apelează rutina BREAK-KEY pentru a verifica dacă nu s-a apăsat tasta de întrerupere a programului. Dacă s-a cerut întrerupere, se sare la rutina de tratare a întreruperii, în caz contrar se încarcă de la variabila de sistem CURCHL, adresa informaţiilor de canal. Se invrementează adresa pînă se ajunge la adresa codului de canal, şi funcţie de codul găsit se sare la programul de tipărire corespunzător. Dacă nu găseşte cod cunoscut, se generează, un raport de eroare.

11.c. Subrutina LFCR

Organigrama — fig. 8A.3.12.c.

Subprogramul, cînd este apelat, trimite codurile de întoarcere ca respectiv salt la linie nouă spre imprimantă (prin rutina IDCH).

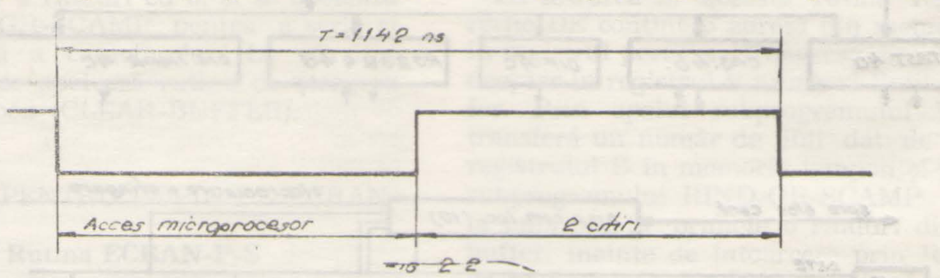


Fig. 2.2

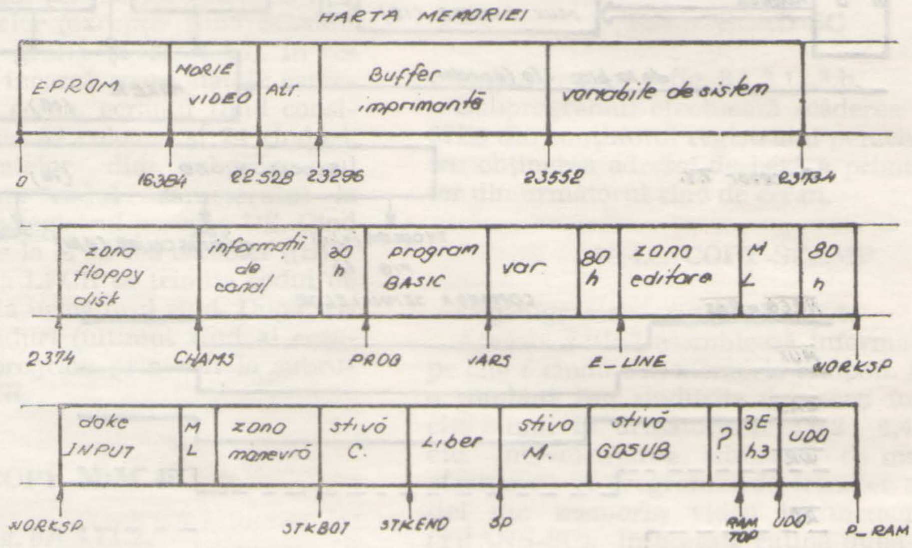


Fig. 2.3

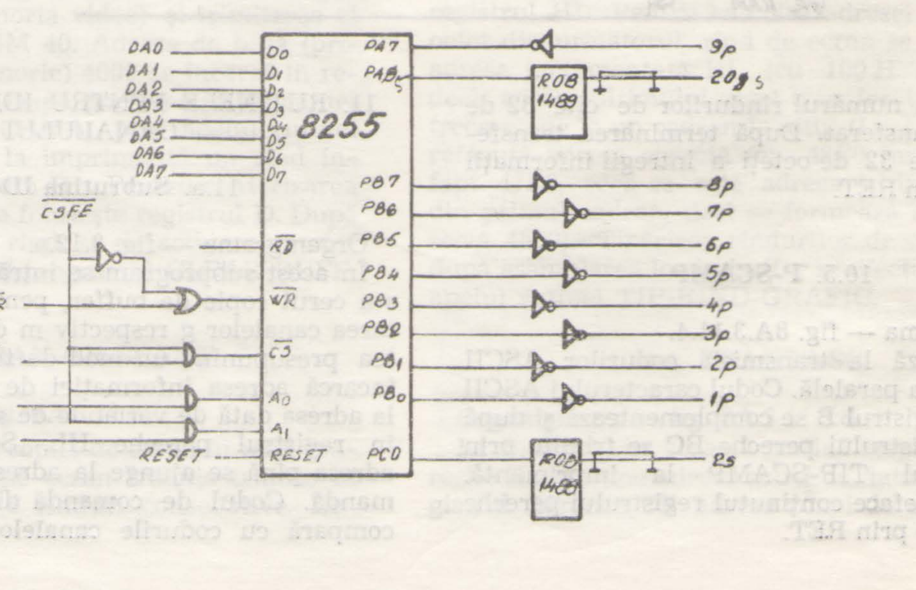


Fig. 2.4.a

B. MANUAL DE UTILIZARE

VII. TASTATURA

Tastatura este formată din taste plate. O tastă poate avea pînă la 5—6 semnificaţii distincte implementate prin intermediul a 256 de caractere pe 8 biţi.

1. De simbol : litera, cifra, semn (ex. A, h, 2, : , #).
2. De simbol compus : cuvinte în BASIC, nume de funcţii sau de constante (ex. DIM, LIST, ABS, PI).
3. De control : ENTER, BREAK, CONT.
4. De simbol semigrafic.
5. De simbol grafic definit de utilizator.
6. De programatoare de taste : CAPS SHIFT, SYMBOL SHIFT.
7. De editor de linii : EDIT, >, V, DELETE.

Pentru realizarea celor maxim 5—6 semnificaţii ale unei taste se folosesc tastele CAPS SHIFT şi SYMBOL SHIFT care pot aduce calculatorul în unul din cele 5 moduri de lucru, care sînt semnalizate prin litera de cursor (care arată unde va fi inserat următorul caracter tastat). Cele 5 moduri au cursoarele K, L, C, E sau G.

MODUL K (simbol compus din interiorul tastei).

Se formează automat la începutul liniei, după ':' sau după THEN. La apăsarea tastei se va edita în linie compozitul sau cifra aflată pe tasta scrisă cu negru. La apăsarea tastei simultan cu tasta SYMBOL SHIFT se va edita simbolul compus sau semnul scris pe tastă cu roşu. Prezenţa cursorului K în linie arată că se aşteaptă un simbol compus sau un simbol care se poate realiza cu el.

MODUL L (litere mici).

Se editează simbolul principal de pe tastă iar la litere se editează litere mici. CAPS SHIFT cu o tastă literă editează litera mare în modul L.

MODUL C (litere mari).

Se editează toate literele în litere mari. CAPS LOCK realizează trecerea L la C şi C la L.

MODUL E (extindere).

Cursorul E apare prin tastarea simultană a CAPS SHIFT şi SYMBOL SHIFT. Este folosit pentru realizarea semnificaţiei scrise deasupra tastei cu albastru prin apăsarea tastei, sau pentru realizarea semnificaţiei scrise dedesubtul tastei cu roşu prin tastarea simultană a tastei alese şi a SYMBOL SHIFT.

MODUL G (semigrafic şi grafic).

Cursorul G apare cu semnificaţia GRAPHICS (tastează 9 şi CAPS SHIFT) şi tot aşa dispare. În acest mod tastarea cifrelor va edita caracterele semigrafice înscrise în taste, iar tastarea literelor de la A la T va edita caractere grafice (necesar) definite de utilizator.

VIII. EDITAREA LINIILOR

Ţinerea apăsată a unei taste mai mult de 1—2 secunde va declanşa editarea semnificaţiei ei în repetiţie.

Editarea liniilor este vizibilă pentru utilizator pe ecranul televizorului. Poziţia cursorului indică unde va continua editarea iar felul cursorului va indica în ce mod se aşteptă continuarea liniei. Liniile se editează pe liniile inferioare ale ecranului alfanumeric. Funcţiile de editare a calculatorului cuplat cu un televizor alb/negru sînt realizate de tastele numerice apăsate simultan cu CAPS SHIFT.

În cursul editării unei linii se poate folosi comanda DELETE care şterge ultimul caracter tastat, comenzile TRUE VIDEO şi INV VIDEO, care vor edita caracterul ce urmează a fi tastat în pozitiv şi negativ, comenzile <,> care vor deplasa cursorul de mod în sensul dorit fără a şterge textul.

După editarea unei linii se va apăsa tasta ENTER. Dacă linia este incorectă va apare un semn ? în linia editată. După ce linia este corectă se apasă ENTER, iar dacă linia nu are număr va fi executată prompt. Dacă linia are număr, ea va fi plasată în textul editat în partea superioară a ecranului.

Ultima linie editată are între număr şi restul liniei marcatorul '<' care este cursorul de linie editată. Dacă se urmăreşte modificarea unei linii anumite se deplasează cursorul linie la linia respectivă cu ajutorul Λ, V apoi se tastează EDIT. Linia va fi adusă în spaţiul de editare unde se modifică tastînd <,> şi DELETE, la terminare tastînd ENTER.

La programare în BASIC e necesar ca liniile programului să aibă un număr (numărul poate fi între 0 şi 9999). Liniile se reprezintă în memorie pe (5+număr de simboluri al liniei) octeţi. În liniile multiinstrucţiune concatenarea instrucţiunilor se face cu ajutorul simbolului ":". Executarea programelor BASIC se face în ordinea crescătoare a numerelor liniilor .

IX. COMENZI

Comanda RESET este implementată cu butonul RESET — cauzează inițializarea sistemului, adică se pierde tot ce a fost înainte ca program sau variabile de program.

Comanda BREAK poate întrerupe un program în execuție și este implementată pe tasta BREAK.

Comanda ENTER, implementată pe tasta ENTER se folosește la introducerea liniilor-program sau a datelor cerute prin instrucțiunea INPUT.

Comanda NEW inițializează sistemul BASIC ștergînd orice program sau variabila anterioară.

Instrucțiunea STOP are aceeași acțiune ca tasta BREAK. Execuția se poate relua cu instrucțiunea CONT (CONTINUE).

Comanda LIST sau LIST n listează pe ecran textul BASIC al programului de la linia n. La umplerea ecranului, în partea de jos a acestuia se afișează "scroll ?", dacă se tastează N listarea liniilor se oprește.

Comanda RUN sau RUN n provoacă inițializarea variabilelor și execuția programului BASIC de la linia n a programului (acțiune identică cu CLEAR și GO TO n).

Comanda GO TO n provoacă execuția programului de la linia n fără a șterge nimic din memorie. Instrucțiunea CLEAR șterge toate variabilele eliberînd spațiul de memorie ocupat de acestea.

Instrucțiunea CLEAR n face același lucru ca și CLEAR ; dacă e posibil modifică variabila de sistem RAMTOP la n și pune stiva GOSUB la adresa n.

X. VARIABILE ȘI CONSTANTE — TABLOURI ȘI ȘIRURI

Reprezentarea numerelor se face pe 9 sau 10 cifre semnificative (deși cu PRINT se editează doar 8 cifre) și este limitată la domeniul 4 * 10e—39 și 10e38. Numerele se memorează cu virgula flotantă, un octet pentru exponent și 4 octeți pentru mantisă.

Numerele întregi mici se reprezintă special, primul octet fiind 0, al doilea fiind semnul (00 sau FF), iar pe următorii doi octeți este întregul în complement de 2.

Numele variabilelor simple pot fi de lungime arbitrară pornind de la o literă și continuînd cu litere și cifre. Lungimea numelui +5 dă numărul de octeți ocupați.

Variabilele din bucle FOR-NEXT și tablouri numerice au nume dintr-o literă.

Numele variabilelor șir și a variabilelor tablou-șir este format dintr-o literă urmată de $.

Tablourile și șirurile pot avea lungimi și număr de dimensiuni și lungime arbitrară.

Toate șirurile într-un tablou de șiruri vor avea aceeași lungime dată de ultima dimensiune din DIM.

Șiruri și tablouri de șiruri nu pot avea același nume.

Atribuirea de noi valori variabilelor în program se face prin instrucțiunea LET.

Numele variabilelor se referă la înțelesul literelor și nu la modul lor de scriere, cu litere mari sau cu litere mici, sau combinat.

Deci AVAS și aVaS desemnează aceeași variabilă.

Tablourile numerice sînt memorate pe un număr de : 4+2 * (numărul dimensiunilor) +5 * (număr de elemente) de octeți.

Șirurile sînt memorate pe un număr de 3+ (lungimea șirului) de octeți. Tablourile de șiruri sînt memorate pe un număr de : 4+2 * (numărul dimensiunilor)+(numărul total de caractere) de octeți.

Rezervarea de spațiu în memorie pentru un tablou sau un tablou șir se face prin utilizarea instrucțiunii DIM.

O variabilă simplă și o variabilă tablou pot avea același nume deoarece sînt distincte ca tip.

La memorarea elementelor de tablou indicele dimensiunii cea mai din stînga se schimbă cel mai încet. Apelarea elementelor unui șir se face prin specificarea caracterului din șir, sau a poziției inițiale și finale a subșirului ce se apelează. Concatenarea șirurilor se face cu "+" între șiruri.

Exemple :

Constante :	3.5e—39, PI, —10.e38, 2, —78.
Variabile :	LET T=2. 171e—15.
	LET INDICE=indice+1.
	LET A8D3BD=DXETD1+ +FG8G.
	LET T2=T1+T0.
	FOR I=1 TO 5 : NEXT I.
	FOR G=822 TO—311 STEP —2 : NEXT G.
Șiruri :	" " —e șirul nul.
	"123ABCXYZ" (4) "A".
	let X$="678XYZ # ()".
	PRINT ; X$ (1 TO 6) arată "678XYZ".
	PRINT : X$ (7) arată "#".
	PRINT ; X$(TO 3), X$ (7 TO) arată "678" și "# ()"
	PRINT ; X$ (4 TO 4) arată "X"
	PRINT : X$ (5 TO 20) dă eroare (sînt doar 9 semne).
Fie :	LET A$="AAA888".
	LET A$ (3 TO 6)="123456".
	PRINT A$ arată "AA1234".
Fie :	LET b$="ABCDEFG".
	LET b$=(2TO7)="XXX".
	PRINT b$ arată "AXXX".

În acest mod se face și atribuirea unei variabile șir sau tablou șir declarată cu DIM (atribuire tip Procust).

Concatenari se pot face prin :

"ABC"+"123"="ABC123".
"ABC" (1 TO 2)+"123" (3TO)= ="AB3".
("ABC"+"123") (3 TO 4)="C1".

Tablouri : DIM A (3) : DIM f (11) : DIM g (7, 8, 2)
DIM Z $ (22) : DIM a $ (3, 7) : DIM B $ (2, 2, 2, 10)
X $ (3)=X $ (3,1)+ ... +X $ 1) : DIM y $ (10, 100)

Fie : DIM X $ (3, 8) tablou şir

Atunci : X $ (1)=X $ (1, 1)+X $ (1, 2)+ ... +X $ (1, 8)
X $ (3)=X $ (3, 1)+ ... +X $ (3, 8)

Fie : LET X $ (2)="ELEMENT"

Atunci : PRINT X $ (2), X $ (2, 4) dă : ELEMENT M

şi : PRINT X $ (2,1 TO 2), X $ (2) (4 TO 7) dă : EL MENT

Declararea dimensiunii şirurilor face ca atribuirea lor să fie de tip Procust.

XI. OPERATORI RELAŢIONALI ŞI LOGICI, EXPRESII

Implementarea expresiilor matematice şi logice în limbajul BASIC se face prin folosirea următorilor operatori :

Logici		Prioritate
OR	(sau logic)	2
AND	(şi logic)	3
NOT	(negare logică)	4
Relaţionali		
=, <, >, <=, >=, <>		5
Aritmetici		
+, —	(adunare şi scădere numere sau concatenare şiruri)	6
*,/	(înmulţire şi împărţire)	8
—	(negare număr)	9
∧	(ridicare la putere)	10
Toate celelalte funcţii		11
Calculul indicilor de tablou, tablou şir sau şir		12

Pentru ridicarea la putere (^) puterea este necesară să fie număr pozitiv, altfel dă eroare B (de exemplu : x^3=(1/x)^3).

La operaţii relaţionale operanzii trebuie să fie de acelaşi tip. Rezultatul e numărul 1 pentru relaţii îndeplinite şi 0 altfel.

Calculul expresiilor poate fi oricît de flexibil folosind parantezele care vor specifica ordinea de operare.

Operatorii logici pot fi consideraţi şi folosiţi ca funcţii :

x AND y --- = x dacă y< >0
= 0 dacă y=0

x OR y .... = 1 dacă y < >0
= x dacă y=0

NOT x --- =0 dacă x < >0
=1 dacă x=0

Iar AND permite şi :

x$ AND y --- =x$ dacă y < >0
= " " (şir nul) dacă y=0

Operatorii relaţionali pot fi folosiţi la şiruri şi se vor referi la caractere din şiruri de operanzi, comparînd codurile caracterelor pe rînd. Deci "ASTA"<"BASTA", "ZYZW">"CALCULATOR" şi "AAA"<"AAB".

Operatorii şi funcţiile de prioritate mai mare (12) sînt evaluaţi înainte de cei de prioritate mai mică (2) în cadrul aceleiaşi expresii.

Exemple :

LET S=s+t.*3/2^n
LET S=s+t.*3/(3^n)
LET FIN=148*72e8*KA^NR—ZE
LET NEG=—NUM
x<>y THEN GO TO 20.
IFx<z AND y>x=w) OR (x<t AND y=v) THEN LET FA=1
IF (x+y/z^t<=ze AND x=F) OR z=t THEN PAUSE 222
IF EXP (V(I)^3+SIN (ALFA (j))) <end AND NOT F THEN STOP

Avem următoarele N expresii adevărate :

"ASTA"<"AIA" (adică "A" vine înainte de "A")

"ASTA">"AIA" (şi "A" vine după "A")

şi "ASTA"<="ASTA" identic sau vine înainte de "ASTA") dar "ASTA"<"ASTA" este expresie falsă şi "ASTA"<"ASTB" este expresie adevărată.

XII. INSTRUCŢIUNI BASIC

Se folosesc următoarele notaţii :

x, y, z — reprezintă expresii numerice
m, n — reprezintă expresii numerice rotunjite la cel mai apropiat întreg
e — reprezintă o expresie
s — reprezintă o succesiune de instrucţiuni separate de :
L — reprezintă o literă
v — reprezintă o variabilă

BEEP x, y

Se folosesc notaţiile engleze pentru notele muzicale, chei, octave.

x reprezintă durata în secunde

y reprezintă înălţimea

Dacă y este pozitiv se iau notele după "do de jos" iar pentru y negativ se iau cele înaintea lui "do de jos" (din octava principală).

Dacă se doreşte schimbarea cheii, cel mai bine este să se introducă o variabilă "cle" care să se adune la înălţimea sunetului :

Exemplu :

BEEP 1, cle+0 : BEEP 5, cle+2 ; BEEP 2, cle+4

Şi pentru durata sunetelor se poate pune variabila.

Exemplu :

BEEP lung, .5 : BEEP lung, 1 : BEEP lung, 3

Gamei DO, RE, MI, FA SOL, LA, SI, DO, îi corespund înălţimile : 0, 2, 4, 5, 7, 9, 11, 12, lui FA diez-6, lui SI bemol-10, lui DO diez-1,

CLS

Şterge imaginea binară generată pe ecranul televizorului. Pune 0 în zona de memorie afectată imaginii binare a ecranului.

DATA el, e2, e3 ...

În instrucţiunile DATA se trec datele ce urmează a fi citite cu instrucţiunea READ. Ordinea de citire a datelor este de la prima instrucţiune DATA şi apoi următoarele pînă la sfîrşitul listei din instrucţiunea READ. Dacă se vrea citirea datelor de la o anumită instrucţiune DATA diferită de prima se foloseşte instrucţiunea RESTORE. Instrucţiunile DATA pot fi plasate oriunde în program.

Exemple :

a) 10 READ d$
20 PRINT "data este :", d$
30 DATA "1 IANUARIE 1986"
40 STOP

b) 10 FOR I=1 TO 8
20 READ A
30 PRINT A
40 NEXT I
50 DATA 10, 2, 15, 99, 54, 22, 73, 29
60 STOP

DEF FN 1 (11, 12,... ln)=e

Cu această instrucţiune utilizatorul poate să-şi definească funcţii proprii în cadrul unui program. Numele funcţiilor trebuie să aibă simbolul "FN" urmate de o literă (dacă rezultatul este un număr) sau "FN" urmate de o literă şi $ (dacă rezultatul este un şir) (deci L sau L $).

Argumentele funcţiei (L 1 L 2 ... Ln) trebuie să apară între paranteze şi pot să fie numerice sau şiruri L sau L i $).

Instrucţiunea DEF se poate plasa oriunde în program.

Exemplu :

10 DEF FNs (x, y)=x+y ; REM suma x+y

Prin x şi y se referă argumentele funcţiei s. După semnul "=", urmează definirea funcţiei prin expresie e.

Odată definită ca funcţie ea se apelează, ca orice funcţie de sistem prin numele ei şi argumente.

Exemplu :

PRINT FNs (2, 4)
PRINT 10+FNs (LEN "floare", 5)

O funcţie poate avea de la 0 la 26 de argumente numerice şi în acelaşi timp 0—26 argumente şir.

DIM L/nl, ..., nh)

sau

DIM L $ (nl, ..., nk)

Foloseşte la declararea dimensiunii tablourilor numerice sau de şiruri. Litera L sau L $ reprezintă numele tabloului iar nl, ..., nk dimensiunile.

La execuţie, instrucţiunea DIM şterge orice alt tablou existent cu litera L sau L $, ia în considerare noul tablou L sau L $ şi îl iniţializează cu valoarea 0 dacă este numeric, sau " " dacă este şir.

Instrucţiunea DIM trebuie să apară în program înainte de utilizarea tablourilor ce apar în această instrucţiune (altfel tablourile sînt considerate nedefinite).

Într-un program pot să apară o variabilă şi un tablou cu aceeaşi literă şi ele nu vor fi confundate întrucît variabila tablou va apare indiciată.

Exemple :

DIM b (10)
DIM c (5, 5)
DIM a $ (5)
DIM h $ (5, 10)

În urma acestui ultim exemplu h$ este considerat ca avînd 5 şiruri de cîte 10 caractere fiecare.

h$ (1, 1) ... h$ (1, 10)
h$ (2, 1) ... h$ (2, 10)
.
.
.
h$ (5, 1) ... h$ (5, 10)

Astfel dacă h $ (2)="1234567890" atunci în urma instrucţiunii PRINT h$ (2), h$ (2,5) vom obţine : 1234567890 şi 5

OBSERVAŢIE

h$ (2,4 TO 8)=h$ (2) (4 TO 8)="45678"

FOR

Are următoarele forme :

FOR i=x TO y

sau

FOR i=x TO y STEP z

Cu instrucţiunea FOR se creează bucle pentru una sau mai multe instrucţiuni ce trebuiesc executate pentru diferite valori ale unor variabile.

i — este contorul de buclă
x — reprezintă valoarea iniţială a contorului
y — reprezintă valoarea finală a contorului
z — reprezintă pasul de trecere de la x la y

La sfîrşitul buclei FOR trebuie să apară instrucţiunea NEXT i (cele 2 litere i din FOR şi NEXT trebuie să coincidă)

Exemplu :

10 DIM b (10)
20 LET S=0
30 FOR i= 1 TO 10
40 READ b (i)
50 LET S=S+b (i)
60 PRINT b (i), S
70 NEXT i
80 DATE 11, 22, 33, 44, 55, 66, 77, 88, 99, 100, 110
90 STOP

Instrucţiunile din cadrul buclei se execută atît timp cît valoarea contorului nu depăşeşte valoarea finală.

Exemplu :

10 LET v=2000
20 FOR i=1000 TO —1000 STEP —50
30 PRINT v—i
40 NEXT i
50 STOP

Două bucle FOR trebuie să fie sau una în interiorul alteia sau complet separate.

Exemplu :

10 FOR m=0 TO 10
20 FOR n=0 TO m
30 PRINT "m=" ; m ; "n=" ; n
40 NEXT n
50 PRINT "Sfîrşit"

```
60 NEXT m
70 STOP
```

Trebuie evitată intrarea în mijlocul unei bucle FOR din afara buclei.

O buclă FOR poate fi scrisă toată într-o singură linie.

Exemplu :

```
FOR i=0 TO 10 : PRINT "i=", i : NEXT i
```

GOSUB n

Se folosește atunci cînd diferite părți de program fac aproape același lucru. Atunci, porțiunile respective se scriu o singură dată de la o anumită linie de program, n ; apelarea subrutinei se face cu instrucțiunea GOSUB n care transferă numărul liniei n în stivă și apoi (la fel ca la GO TO) se face un salt la linia n. Se execută instrucțiunile după linia n pînă la întîlnirea instrucțiunii RETURN cînd programul revine la prima instrucțiune după instrucțiunea GOSUB.

Instrucțiunea RETURN este ultima instrucțiune dintr-o subrutină și este obligatorie.

Exemplu :

```
10 CLS
20 INPUT x
30 INPUT y
40 REM
50 IF x<0 THEN GOSUB 100 : GO TO 20
60 IF y<0 THEN GOSUB 100 : GO TO 20
70 GOSUB 200
80 PRINT FNs (x, y)
90 GO TO 20
100 PRINT "Eroare"
110 RETURN
200 DEF FNs (x, y)=SQR x+SQR y
210 RETURN
```

GO TO n

Cînd e întîlnită această instrucțiune se execută un salt la linia cu numărul n (sau la prima după ea dacă linia n nu există) și se execută în continuare programul de la această linie.

Exemplu :

```
10 INPUT a
20 IF a<0 THEN GO TO 10
30 IF a=0 THEN GO TO 50
40 PRINT a
50 STOP
```

IF e THEN s

Instrucțiunea IF este instrucțiune de salt condiționat de valoarea de adevărat sau fals a expresiei e.

Expresia e poate conține variabile (numerice sau șir) constante, operatori relaționali (=, <>, <, >, <=, >=) operatori (funcții) logici (AND, OR, NOT) și funcții ale sistemului sau definite de utilizator.

De exemplu 1<2 sau —5<—2 sînt adevărate iar 1<0 sau 0<—2 sînt false.

s poate fi o instrucțiune GO TO, o altă instrucțiune sau o succesiune de instrucțiuni.

Exemplu :

```
IF a<b THEN GO TO 50
IF a $="DA" AND x>0 THEN PRINT x : STOP
IF NOT x=y THEN STOP
```

Dacă s este format din concatenari de instrucțiuni se recomandă atenție la compoziția lui s.

INPUT . . .

Prin . . . am semnificat o succesiune de variabile numerice sau șir separate prin virgulă sau LINE.

Se folosește pentru introducerea datelor de la tastatură. La întîlnirea instrucțiunii calculatorul se oprește din execuție și așteaptă introducerea datelor.

Exemplu :

```
INPUT a
INPUT "TEXT", a ; AT 1, 1 : "LINIA 1" ; AT 2, 1 : "LINIA 2"
INPUT "Introduceți valoarea lui a", a
INPUT a $
INPUT LINE a $
```

Diferența dintre ultimele două exemple este următoarea : pentru INPUT a $, calculatorul afișează" și așteaptă introducerea caracterelor după care pune ghilimelele de sfîrșit ; pentru INPUT LINE a $ nu se mai afișează, .utilizatorul introduce caracterele de intrare asociate variabilei șir a $ de exemplu : Azi, iar variabila a $ va avea valoarea Azi. Întrucît " nu apar pe șirul a $ el nu pot fi șterse sau utilizate în expresii.

LINE nu poate fi utilizat pentru variabile numerice.

LET v=e

Este o instrucțiune de atribuire ; atribuie valoarea expresiei e variabilei v. O variabilă simplă (neindiciată) este considerată nedefinită pînă cînd nu apare dintr-o instrucțiune LET, READ sau INPUT. Dacă v este o variabilă șir indiciată atunci valoarea lui e este sau trunchiată sau umplută cu spații pînă la dimensionarea variabilei v.

Exemple :

```
LET a=10 : LET b (i)=i*x+5
LET a $="text"
LET t $="TEXTUL" : LET V $="AM"
INPUT (T$)' '"SCRIS" ; (V$) ; W$
```

Caracterul ' face să fie sărită o linie. Variabila T$ și V$ între paranteze se editează pe ecran, iar W$ se citește.

LIST sau LIST n

Prima formă este echivalentă cu LIST 0.

Instrucțiunea listează 22 de linii din programul existent în calculator de la instrucțiunea cu numărul cel puțin n, începînd din partea de sus a ecranului și face linia n linie curentă, apoi afișează "scroll ?". Pentru 'N' oprește afișarea.

PAUSE n

Oprește execuția programului și așteaptă un număr de n/50 secunde ; n trebuie să fie cuprins între 0 și 65535. Pentru PAUSE 0 calculatorul se oprește de tot din execuție iar 65535 corespunde la aproximativ 22 minute. Pentru continuare la oprirea cu PAUSE se tastează orice tastă.

PRINT ...

Prin ... am specificat o succesiune de variabile sau expresii numerice sau şir separate de virgulă, punct şi virgulă sau apostrof.

Dacă se tipăreşte o expresie sau variabila numerică mai întîi se tipăreşte minus dacă valoarea este negativă. Dacă valoarea este <=10e—5 sau >=10e—13 atunci este tipărită cu mantisa exponent (E+(—)). În restul cazurilor valoarea este tipărită în notaţie zecimală obişnuită.

Pentru variabile tip şir se tipăresc caracterele ţinîndu-se cont de caracterele de control. Caracterul ; nu are nici un efect special. valorile sînt tipărite una după alta pe aceeaşi linie.

Caracterul , face ca tipărirea valorilor să se facă cîte 2 pe linie, una la începutul liniei iar celălalt la mijloc.

Caracterul ' (apostrof) face ca valorile să fie tipărite una sub alta.

Exemple :

```
PRINT 5 ;2
```

execută următoarele : 52

```
PRINT 5,2
5       2
PRINT 5'2
5
2
```

În instructiunea PRINT se poate specifica de exemplu linia (0—21) şi coloana (0—31) unde dorim să se facă afişarea valorilor folosind

AT m, n ;

Exemplu :

```
PRINT AT 10,5 ; "m=7"
```

va tipări pe linia 10 incepînd din coloana 5 caracterele m=7. Sau PRINT AT 10,5 ; "m=" ; AT 10,7 ; 7

TAB n

unde n este un număr de coloană unde se mută poziţia de tipărire, pe aceeaşi linie sau următoarea dacă sînt necesare spaţii înapoi de la poziţia curentă.

Calculatorul reduce numărul coloanei modulo 32 (se împarte la 32 şi se ia restul) ; deci TAB 33 este echivalent cu TAB 1.

Exemple :

```
PRINT AT 5,1 ; "INTRODUCERE" ; TAB 26 ; 10
```

Cu instrucţiunea PRINT pot fi folosite instrucţiunile OVER şi INVERSE cu aceleaşi funcţii ca cele descrise la instrucţiunile grafice.

Exemple :

```
OVER 0
FOR i=1 TO 20
PRINT AT 5,5 ; "test"
PAUSE 11
PRINT AT 5,5 ; "prog"
PAUSE 11
NEXT i
```

sau

```
OVER 1
FOR i=1 TO 32
PRINT "u" ; CHR$ 8 ; "'"
NEXT i
```

RANDOMIZE sau RANDOMIZE n

Prima formă este echivalentă cu RANDOMIZE 0.

Apare ca şi RAND pe tastatura calculatorului. Se foloseşte împreună cu instrucţiunea RND pentru care generează o valoare de start n, ce poate fi cuprinsă între 1 şi 65535.

Dacă n=0 atunci valoarea de start pentru RND este aleasă în funcţie de timpul de cînd calculatorul este pornit.

Exemplu :

```
RANDOMIZE 5
```

READ y1, y2 ...

Atribuie valorilor y1, y2 ... expresii succesive din listele instrucţiunilor DATA. Variabilele sînt separate prin virgulă.

Exemplu :

```
10 DIM b(10)
20 FOR i=1 TO 10
30 READ b(i)
40 PRINT b(i)
50 NEXT i
60 READ a$, h$
70 PRINT a$, h$
80 DATA 10, 11, 12, 13, 14, 15, 16, 17, 18, 19
90 DATA "azi este", "joi"
100 STOP
```

Citirea începe cu prima instrucţiune DATA din program. Astfel b(1)=10, b(2)=11 ... b(10)=19, apoi a$="azi este" şi h$="joi".

REM' ... '

' ... ' poate fi orice succesiune de caractere cu excepţia lui ENTER. Se foloseşte la realizarea de comentarii la programe. Într-o linie de program, după instrucţiunea REM nu are sens să apară nici o altă instrucţiune.

Exemplu :

```
REM ' secţiune de iniţializare '
```

REM poate fi folosit ca şi comandă. Astfel după instrucţiunea LIST urmată de ENTER dacă la întrebarea "scroll ?" se răspunde N apare mesajul "D BREAK-CONT repeats". Dacă se apasă din nou ENTER se vor vedea 22 linii din program.

Dacă se introduce de exemplu '30 REM' se vor lista liniile 9 la 30, pentru '40 REM' se vor lista liniile 19 la 40.

RESTORE sau RESTORE n

Prima formă este echivalentă cu RESTORE 0.

Restabileşte pointerul DATA la prima instrucţiune DATA în linia cu numărul cel puţin n ; următoarea instrucţiune READ va începe citirea de la această instrucţiune DATA.

Exemplu :

```
10 DIM b(5)
20 FOR i=1 TO 5 : READ b(i) : PRINT b(i)
NEXT i
30 RESTORE 60
40 READ x, y, z
50 DATA 15,22
60 REM "date comune"
70 DATA 10, 20, 30
80 STOP
```

Astfel b(1)=15, b(2)=22, b(3)=10, b(4)=20, b(5)=30

Instrucțiunea RESTORE 60 restabilește pointerul la 60 DATA și deci x, y, z se vor citi în această listă : x=10, y=20, z=30.

RND

Dă următorul număr pseudoaleator într-o secvență generată prin luarea puterilor lui 75 modulo 65537, scăzînd 1 și împărțind prin 65536. Numărul generat e cuprins între 0 și 1 (uneori poate fi 0 dar niciodată 1). Cu RND se pot genera numere pseudoaleatoare în orice domeniu (a, b) cu formula :

(b—a) * RND+a

Exemplu :

```
10 FOR n=1 TO 100
20 PRINT 5+RND * 7
30 NEXT n
```

Va tipări numere reale pseudoaleatoare în intervalul (5, 12).

STOP

Oprește programul cu eroare 9. Se poate continua programul tastînd comanda CONTINUE.

XIII. INSTRUCȚIUNI GRAFICE

Partea din ecran utilizabilă conține 256×176 pixeli direct adresabili. Pixelul de adresă 0,0 se află în colțul din stînga jos al ecranului grafic. Ecranul alfanumeric conține 24 de linii a cîte 32 coloane (caractere) caracterul de adresă alfanumerică 0,0 fiind în colțul din stînga sus. Liniile 22 și 23 sînt rezervate mesajelor de la sistem. Un caracter alfanumeric este reprezentat pe o matrice de 8×8 pixeli.

Instrucțiunile grafice sînt : PLOT, DRAW, CIRCLE și POINT.

PLOT

PLOT m ; x, y

m — reprezintă una din instrucțiunile OVER sau INVERSE

x, y — reprezintă coordonatele pe x respectiv y

PLOT x, y

Mută cursorul grafic din poziția curentă la poziția de coordonate x, y și marchează pixelul respectiv scrie.

PLOT INVERSE 1 ; x, y

La fel ca mai sus dar șterge punctul x, y

PLOT OVER 1 ; x, y

La fel ca mai sus dar dacă punctul a fost scris îl șterge și dacă a fost nescris îl scrie (SAU EXCLUSIV între cele două puncte).

PLOT INVERSE 1 ; OVER 1 ; X, Y

Lasă pixelul neschimbat și cursorul grafic rămîne poziționat în x, y

DRAW

DRAW x, y, z

DRAW x, y (identic cu DRAW x, y, 0)

Trage o linie de la poziția curentă a cursorului grafic pînă la poziția incrementală cu x pixeli pe orizontală și y pixeli pe verticală — z reprezintă unghiul în radiani pe care trebuie să-l aibă segmentul de cerc care unește cele 2 puncte.

Dacă z este pozitiv curba se întoarce spre stînga, dacă este zero se trage o linie dreaptă între cele două puncte.

DRAW OVER 1 ; x, y, z

Face un SAU EXCLUSIV între traseul existent pe ecran și o linie scrisă definită de parametrii instrucțiunii (x, y, z).

DRAW INVERSE 1 ; x, y, z

Șterge linia pe traseul definit de x, y, z.

CIRCLE

CIRCLE x, y, z

Trasează un cerc cu centrul în pixelul de coordonate x, y, de raza z pixeli.

POINT

POINT (x, y)

Funcția POINT are valoarea 1 sau 0 după cum pixelul de coordonate x, y este scris sau nescris.

Exemple :

```
PLOT 11, 22
PLOT INVERSE 1 ; 0,0
PLOT OVER 1 ; 50, 50
PLOT INVERSE 1 ; 33, 44
DRAW 30, 30
DRAW — 10, 11
DRAW INVERSE 1 ; 0, 35, PI/3
DRAW OVER 1 ; —200, 0.
DRAW 25, 25, PI
DRAW —7, 18, —PI/5
CIRCLE 100, 100, 25
CIRCLE 175, 100, 75
PRINT POINT (0, 0)
LET x=POINT (100, 121).
```

XIV. PROGRAMAREA CULORILOR

Calculatorul poate fi programat astfel încît interacțiunea utilizator-televizor color are (pe lîngă transferul informației alfanumerică și grafică) posibilitatea transferului informației prin culoare.

Sînt disponibile 8 culori :

0 — negru
1 — albastru
2 — roșu
3 — violet
4 — verde
5 — albastru deschis
6 — galben
7 — alb

Culorile sînt programabile atît utilizînd instrucțiunile cît și prin tastarea tastelor numerice 0—7. Pe un televizor alb/negru în locul culorilor apar nuanțe gri, cifra cod (0—7) fiind ordonată după strălucirea nuanțelor de gri.

Imaginea de pe ecranul televizorului color este formată din 768 de poziții caracter (24 de

linii a 32 de coloane). Fiecare poziție-caracter este caracterizată de :

a) o matrice de 8×8 puncte-culoare numită formă binară avînd 1 pe punctele "scrise" cu cerneală (INK) și 0 în punctele "nescrise" (PAPER)

b) culorile "hîrtiei nescrise" (a fondului) PAPER 0 . . . 7

c) culoarea "scrisului" (a cernelii) INK 0 . . . 7

d) strălucirea poziției caracter 0-normal, 1-strălucitor
BRIGHT 0 sau 1

e) clipirea poziției-caracter 0-fără, 1-clipitor
FLASH 0 sau 1 (1-activ, 0-inactiv)

Caracteristicile b, c, d și e sînt numite atribute ale poziției caracter. Se remarcă faptul că într-o poziție-caracter pot exista maximum 2 culori (INK și PAPER — cerneală și hîrtie), iar atributele BRIGHT și FLASH se referă la o poziție-caracter completă. Prin printare pe o poziție-caracter anumită se pot schimba caracteristicile a la e.

Instrucțiunile de culoare PAPER, INK, BRIGHT, FLASH pot avea argument pe lîngă valorile date mai sus și cifra 8 cu semnificația de transparență și rezultatul ca prin printarea cu noile atribute, rămîn vizibile prin transparență vechile atribute (de exemplu se poate păstra culoarea veche a hîrtiei sau a cernelii, sau vechea lucire sau clipire).

Instrucțiunile PAPER și INK pot avea și argumentul 9 ceea ce înseamnă "contrast". Sînt considerate culori închise : negru, albastru, roșu și violet și culori deschise : verde, albastru deschis, galben și alb. INK 9 (PAPER 9) provoacă scrierea cu cerneală albă (pe un fond alb) dacă culoarea de fond (cerneală) a fost închisă și scrierea cu cerneală neagră (pe fond negru) dacă a fost culoarea deschisă de fond (cerneală deschisă).

Atributele unei poziții-caracter de pe ecran pot fi aflate folosind funcția

ATTR (linie, coloană)

argumentele funcției fiind aceleași cu cele folosite în funcția AT. Rezultatul funcției este un octet (număr între 0 și 255) care va fi interpretat astfel :

— bit 7 pentru FLASH (0 sau 1)
— bit 6 pentru BRIGHT (0 sau 1)
— biții 5-4-3 pentru PAPER
— biții 2-1-0 pentru INK

Astfel numărul rezultat este o sumă formată din :

128 pentru FLASH 1 | 0 pentru FLASH 0
64 pentru BRIGHT 1 | 0 pentru BRIGHT 0
8 * Codul hîrtiei
Codul cernelii

Controlul formei binare 8×8 aflată la o poziție-caracter (caracteristica a) este realizat cu instrucțiunile INVERSE și OVER care au ca argumente 0 sau 1 (0-inactiv, 1-activ). INVERSE 1 provoacă schimbarea punctelor scrise în puncte nescrise și invers. Instrucțiunea OVER 1 permite vizualizarea unei forme binare suprapusă peste altă formă binară făcînd SAU EXCLUSIV între cele 2 forme binare (punct cu punct).

Toate instrucțiunile referitoare la caracteristicile poziție-caracter pot fi folosite ca părți ale instrucțiunii PRINT și atunci au o acțiune locală afectînd în execuție doar instrucțiunea PRINT asociată.

Instrucțiunea BORDER n (n=0 . . . 7) programează culoarea restului de ecran TV și a părții inferioare (2 linii). Pe liniile inferioare de ecran, unde se transmit datele cu INPUT și unde sistemul dă mesaje, obișnuit culoarea hîrtiei (PAPER) este cea a restului de ecran (BORDER) iar culoarea cernelii este contrast (INK 9) și nu există lucire sau sclipire (BRIGHT 0 și FLASH 0).

Prin definirea explicită în instrucțiunea INPUT (la fel ca la PRINT) se pot schimba aceste atribute, însă acțiunea lor este valabilă doar cît se poate executa instrucțiunea INPUT sau pînă sînt introduse date de intrare.

Schimbarea atributelor de culoare prin tastare la editarea programelor se face prin stabilirea modului E de editare (simultan CAPS SHIFT și SIMBOL SHIFT) apoi :

— tastele : 0—7 dau culoarea hîrtiei (PAPER)

— CAPS SHIFT și tastele 0—7 dau culoarea cernelii (INK)

— tastele 8 și 9 dau strălucirea : BRIGHT 0 respectiv BRIGHT 1

— CAPS SHIFT și tastele 8 și 9 dau clipirea : FLASH 0 respectiv FLASH 1

În modurile K, L sau C, CAPS SHIFT și tastele 3 și 4 realizează TRUE VIDEO și INVERSE VIDEO pentru caracterele editate adică trecerea în editare video normală sau inversată (similar cu INVERSE 1).

Introducerea atributelor prin tastare provoacă generarea în textul editat a caracterelor FLASH și BRIGHT vizibile doar prin acțiunea lor. Tastînd DELETE se șterg și aceste caractere invizibile din textul de editare, chiar dacă apar efecte secundare.

Exemple :

Încercați pentru testare următoarele programe :

```
10 CLS : FOR i=1 TO 75
20 BORDER 1 : INK RND * 7
30 PAPER RND * 7
40 PRINT "TIM-S"
50 NEXT i
```

```
10 BORDER 0 : CLS : PAPER 1
20 LET c=4
30 FOR x=1 TO 12
40 READ t
50 FOR 1=21 TO 21-t STEP-1
60 PRINT PAPER 6 ; AT 1, C ; " "
70 NEXT 1
80 PRINT INK 2 ; AT 20-t, c ; t
90 LET c=c+2
100 NEXT x
110 DATA 20, 15, 13, 16, 19, 20, 18, 11, 12,
19, 14, 17
```

```
10 BORDER 0 : CLS : PAPER 0
20 FOR x=7 TO 0 STEP-1
30 INK x
40 FOR 1=11—x TO 11+x
50 FOR c=16—x TO 16+x
60 PRINT AT 1, c ; "* "
70 NEXT c
80 NEXT 1
90 NEXT x
```

Pentru testarea instrucțiunii INVERSE adăugați :

```
15 INVERSE 1
```

iar pentru testarea instrucțiunilor BRIGHT și FLASH încercați cu :

```
15 BRIGHT 1
16 FLASH 1
```

Ca exemplu pentru umplerea figurilor încercați :

```
10 BORDER 1 : CLS : PAPER 6 : INK 2
20 FOR x=100 TO 100
30 PLOT 128, 150
40 DRAW x, —120
50 NEXT x
```

și puteți obține diferite efecte adăugînd în linia 20 la instrucțiunea FOR opțiunea STEP :

```
20 FOR x=—100 TO 100 STEP 4
```

Următorul program schimbă atributele aleator, direct în memoria ecran :

```
10 POKE 22527+RND * 704, RND * 127
20 GO TO 10.
```

XV. INSTRUCȚIUNI PENTRU LUCRUL CU CASETA

Casetele cu care se lucrează sînt casete audio care pentru o lungime de 60 minute asigură un spațiu de înregistrare de circa 900 de Kocteți (450+450).

Numele fișierelor pot avea maxim 10 caractere. Se pot manevra și citi de pe casetă patru tipuri de informații : program BASIC împreună cu variabilele, tablouri numerice, tablouri de caractere și șiruri de biți (program în limbaj de asamblare sau imagini binare).

Instrucțiunile pentru lucru cu caseta sînt : LOAD, MERGE, SAVE, VERIFY.

LOAD

Înainte de execuția instrucțiunii se șterge vechiul program aflat în calculator împreună cu variabilele.

LOAD "nume"

Încarcă de pe casetă fișierul "nume".

LOAD ""

Încarcă primul fișier întîlnit.

LOAD "nume" DATA litera ( )
LOAD "nume" DATA litera $ ( )

Șterge orice tablou cu numele literă sau litera $ din programul aflat în calculator și încarcă în loc ce găsește în fișierul "nume" de pe casetă.

LOAD "nume" CODE
LOAD "nume" CODE "start"
LOAD "nume" CODE "start", "lungime"

Încarcă un șir de biți din fișierul "nume" în memoria calculatorului începînd de la adresa "start" (pe un număr de octeți "lungime") scriind peste ce găsește în memorie.

LOAD "nume" SCREEN $

Sinonim cu LOAD "nume" CODE 16384, 6912 care realizează încărcarea în zona de memorie alocată imaginii TV, a fișierului "nume".

MERGE

MERGE "nume"

Execută același lucru ca și LOAD cu diferența că șterge din liniile și variabilele vechiului program aflat în memorie, doar cele care se suprapun ca număr sau ca nume cu cele din programul aflat în fișierul "nume".

Fișierul "nume" conține numai programe în limbajul BASIC.

SAVE

Salvează pe casetă diverse fișiere, identificate prin nume adică așteaptă pornirea casetofonului pentru înregistrare și apăsarea oricărei taste, după care se execută memorarea audibilă în difuzorul casetofonului. La terminarea în bune condiții a înregistrării apare mesajul O.K.

SAVE "nume"

Salvează programul BASIC și variabilele lui în fișierul "nume".

SAVE "nume" LINE n.

La fel ca la forma anterioară doar că la încărcare programul intră automat în execuție de la linia n.

SAVE "nume" CODE "start", "lungime".

Memorează șirul de biți de la adresa "start" pe "lungime" octeți în fișierul "nume".

SAVE "nume" DATA litera ( )
SAVE "nume" DATA litera $ ( )

Memorează tablouri numerice "litera" sau șir "litera $" în fișierul "nume".

SAVE "nume" SCREEN $

Salvează imaginea binară aflată pe ecranul TV în fișierul "nume".

VERIFY

Verifică informațiile de pe casetă cu informațiile aflate deja în memorie.

Are aceeași formă și semnificație ca instrucțiunea LOAD doar că nu se face o încărcare în memorie ci o comparare cu memoria.

Dacă informațiile sînt identice se dă mesajul O.K. iar în caz de neconcordanță se dă mesajul de eroare "R TAPE LOADING ERROR". De obicei se folosește după SAVE pentru verificarea corectitudinii memorării pe casetă, întrucît unele casete magnetice prezintă defecte în pelicula magnetică (foarte rar).

Exemple :

```
LOAD "PROG1"
LOAD "BIN1"
LOAD "Binar" CODE 30000
LOAD ""
LOAD "FIS120" CODE 23230, 2422
LOAD "MAT B" DATA B ()
```

LOAD "text" DATA F$ ( )
LOAD "FIGURA" SCREEN$
MERGE "SUPRAPUS"

Exemplele sînt valabile şi pentru instrucţiunile SAVE şi VERIFY.

XVI. FUNCŢII

Notaţii :

x, y — variabile numerice
x$ — şir de caractere

ABS X

Rezultatul este valoarea absolută a lui x.

Exemplu :

ABS —3.2 = ABS 3.2 = 3.2.

ACS x

x cuprins între —1 şi 1. Rezultatul este în radiani şi reprezintă funcţia arccosinus (x).

Dacă x nu este în domeniul apare mesaj de eroare A.

x AND y

y este întotdeauna un număr ; x poate fi număr sau şir de caractere. Rezultatul este următorul :

— dacă x este un număr :

x AND y = 0 (fals) pentru y = 0 (fals)
x pentru y ◊ 0 (y adevărat)

— dacă x este şir de caractere :

x$ AND y = x$ pentru y ◊ 0
"" pentru y = 0

ASN x

Rezultatul este funcţia arcsinus (x). Dacă x nu este cuprins între —1 şi 1 sistemul dă mesajul de eroare A.

ATN x

Rezultatul este funcţia arctangenta (x) în radiani.

CHR $ x

Rezultatul este caracterul al cărui cod este x, cu x rotunjit la cel mai mic întreg.

Exemple :

CHR$ 36 = "$"
CHR$ 87 = "W"

CODE x$

x$ este şir de caractere. Funcţia dă codul primului caracter din şirul x$ sau 0 dacă x$ este şirul gol.

COS x

x este un număr în radiani. Rezultatul este funcţia cosinus (x).

EXP x

Rezultatul este funcţia exponenţială e la puterea x.

INT x

Rezultatul este partea întreagă a lui x (întotdeauna rotunjită în jos).

Exemple :

INT 3.9 = 3
INT —3.9 = —4

LEN x$

Rezultatul este lungimea şirului x$ (numărul de caractere al şirului).

Exemplu :

PRINT LEN "exemplu" va tipări numărul 7.

Dacă într-o singură expresie apar şi funcţii şi operaţii atunci funcţiile vor fi evaluate înaintea operaţiilor. Pentru siguranţă se vor folosi paranteze.

Exemple :

LEN "prog" + LEN "ramul" va fi evaluată astfel :

4 + LEN"ramul"
4 + 5
9

iar expresia LEN ("prog" + "ramul") va fi evaluată astfel :

LEN ("programul")
LEN "programul"
9

LN x

x este un număr >0. Rezultatul este logaritmul natural al lui x, adică în baza e (inversa funcţie exponenţială). Dacă x<0 apare mesajul de eroare A. Pentru obţinerea logaritmului în altă bază se foloseşte formula de transformare :

$\log_a x = LN\ x/LN\ a$

NOT x

Rezultatul este 0 (fals) dacă x<>0 (adevărat) şi 1 (adevărat) dacă x = 0 (fals). Funcţia NOT are prioritate 4.

x OR y

x şi y sînt numere. Rezultatul este operaţia binară "sau". Valoarea operaţiei este 1 (adevărat) dacă y◊0 (adevărat) şi x dacă y = 0 (fals).

PI

Nu are argument. Dă numărul 3,14159265...

SGN x

Rezultatul este semnul lui x ; este —1 pentru x<0, 0 pentru x = 0 şi +1 pentru x>0.

SIN x

x este în radiani. Rezultatul este funcţia sinus (x).

SQR x

x este un număr >0. Rezultatul este radical de ordin 2 (rădăcina patrată) din x. Dacă x<0 apare mesajul "AN INVALID ARGUMENT".

Exemple :

SQR 4 = 2
SQR 0.25 = 0.5

STR $ x

Converteşte numere în şiruri, deci rezultatul este un şir format din cifrele numărului x.

Exemplu :
LET a$=STR$ 1e2
este identic cu :
LET a$=STR$ 100
şi a$ va fi "100".

TAN x

x este un număr în radiani. Rezultatul este funcţia tangentă (x).

VAL x$

Într-un anume sens VAL este funcţia inversă pentru STR $. Converteşte şiruri de numere, deci rezultatul este un număr.

Exemplu :
VAL "3.5" = 3.5.

Dacă se ia un număr şi se aplică succesiv STR $ şi VAL, se va obţine acelaşi număr. Dacă însă se ia un şir, se aplică VAL şi STR$, nu se obţine întotdeauna acelaşi şir, aceasta deoarece şirul argument al funcţiei VAL poate fi orice expresie numerică şi va fi evaluată corespunzător de funcţia VAL.

Exemplu :
VAL ("2"+"* 3") va fi evaluată astfel :
VAL ("2 * 3").
2 * 3

OBSERVAŢII :

În interiorul unui şir de caractere, de cîte ori apar "" (deci şir în interiorul altui şir) numărul de ghilimele trebuie dublat (din 1 în 2, din 2 în 4 etc.).

Exemplu :
PRINT VAL "VAL" "VAL"" "" 2"""""""
Dacă x$ conţine erori de sintaxa apare mesaj de eroare C.

VAL$ x$

Se evaluează x$ fără ghilimelele de margine iar ce rămîne este evaluat tot ca un şir. Deci aici şi argumentul şi rezultatul funcţiei este tot un şir.

Exemplu :
VAL $"""" Mesaj """" = "Mesaj"

XVII. ALTE INSTRUCŢIUNI ŞI FUNCŢII

POKE m, n şi PEEK m

POKE înscrie valoarea n octetului de memorie de la adresa m. Dacă nu avem 0<=m<=65535 şi 0<=n<=255 dă eroare B.

Această înscriere se face fără intermediul mecanismelor folosite normal de BASIC.

Opusa instrucţiunii POKE este funcţia PEEK care citeşte din memorie octetul de la adresa m şi atribuie valoarea înscrisă în el funcţiei.

OUT m,n şi IN m

Analog cu celulele de memorie există şi porturi I/0 cu adrese de la 0 la 65535, porturi de intrare/ieşire. Aceste porturi sînt utilizate de procesor pentru lucrul cu exteriorul (de ex. tastatura, printer, casetofon etc.) şi pot fi citite şi înscrise în acelaşi mod ca memoria cu :
OUT m,n şi IN m

Aceleaşi semnificaţii pentru m şi n.

IN este funcţia sinonimă cu PEEK şi OUT este instrucţiune sinonimă cu POKE.

Adresele memoriei (sau I/O) sînt exprimate în binar pe 16 biţi, de la 0 la 15, bitul 15 fiind cel mai semnificativ.

Exemple :

1. Pentru POKE 52236, 253
PRINT PEEK 52236 arată 253

2. Pentru a face calculatorul să nu întrebe "scroll ?" se va înscrie :
POKE 23692, 255
şi calculatorul va edita pe ecran 255 * 22 de linii în mod continuu.

3. Se poate vedea conţinutul registrului B al procesorului prin PRINT PEEK 23655.

4. Tonul de tastare se schimbă cu :
POKE 23609, n
Lungimea tonului de tastare prin :
POKE 23608, n
BIN . . . .

5. Tastatura este formată din 8 jumătăţi de linii de cîte 5 taste.

IN 65278	citeşte CAPS SHIFT	la V
IN 65022	"	A la G
IN 64510	"	Q la T
IN 63486	"	1 la 5 etc.
IN 32766	"	SPACE LA B.

Adresele sînt formate : 254+256 *(255—2 ^ n)
unde n=0 . . . 7.

Cu IN se citeşte un octet. Biţii de la 0 la 4 pe 1 arată că cele 5 taste sînt neapăsate (bitul 0 pentru tasta exterioară).

6. Portul 254 cu OUT conduce difuzorul pe bitul 4 şi mufa de înregistrare pe bitul 3.

Numărul zecimal corespunzător unei configuraţii binare pe cel mult 8 biţi se află cu funcţia BIN (BIN 10011100=56).

Pentru memorarea simbolurilor grafice definite de utilizator UDG se aminteşte că un caracter are 8×8 pixeli (puncte luminoase).

În această matrice 8×8 se poate defini orice configuraţie de puncte stinse (0) şi aprinse (1).

Considerăm configuraţia :

Linia	1	1 1 1 1 1 1 1 1
	2	1 1 0 0 0 0 1 1
	3	1 0 1 1 1 1 0 1
	4	1 0 1 0 0 1 0 1
	5	1 0 1 0 0 1 0 1
	6	1 0 1 1 1 1 0 1
	7	1 1 0 0 0 0 1 1
	8	1 1 1 1 1 1 1 1

Presupunem că vrem să memorăm această formă binară în locul semnului UDG M. Adresa semnului M se află cu funcţia : USR "M".

Pentru a înscrie simbolul definit vom scrie :
POKE USR "M" +i, BIN (linia i).
unde i=0 la 7 este numărul liniei.

După aceasta în modulul G ori de cîte ori vom folosi tasta "M" se va imprima pe ecran semnul

definit mai sus. Aceeaşi remarcă pentru printarea caracterului de cod ASCII 156.

PRINT CHR $ 156

Se pot folosi pentru caractere grafice definite de utilizator simbolurile cu cifrele de cod 144—164.

Caracterele cu codurile 128—143 sînt simboluri semigrafice.

Exemple :

5=BIN 101
170=BIN 10101010
255=BIN 11111111
16=BIN 10000

LLIST, LPRINT

Aceste instrucţiuni se referă la imprimantă (care poate fi cuplată cu calculatorul într-o configuraţie extinsă), şi au aceleaşi funcţii ca LIST şi PRINT (care se referă la TV) şi aceeaşi sintaxă. În cazul acestor instrucţiuni listarea sau printarea la imprimantă va decurge continuu pînă la terminarea instrucţiunilor ; mesajul "scroll" nu va mai apare.

Imprimanta poate fi oprită prin tastarea lui BREAK.

Funcţiile AT şi TAB vor acţiona diferit la folosirea în LPRINT. Liniile sînt transmise spre imprimantă prin intermediul unei memori tampon pe măsură ce se execută programul. Deci AT îşi pierde sensul, şi nu va putea cauza printarea unei linii (aceasta se va face automat cînd registrul tampon este plin). Atributul TAB în schimb, va cauza printarea liniei dar caracterele vor fi puse din coloana 1.

COPY

Instrucţiunea cauzează realizarea unei copii a imaginii binare a ecranului pe imprimanta grafică corespunzătoare.

INKEY $

Este o funcţie fără argument care citeşte tastatura în momentul cînd este executată în program. Rezultatul este un caracter în modul L sau C, dacă se apasă o tastă, sau şirul vid ; astfel :

IF INKEY $="A" THEN GO TO 200
20 IF INKEY $=" " THEN BEEP 1,3 : GO TO 20
sau
LET A $=INKEY $
IF CODE (A $)>64 THEN PRINT A $

POINT (x, y)

Funcţia are valoarea 1 dacă pixelul de coordonate x, y este scris şi 0 altfel.

Se foloseşte pentru interacţiunea cu imaginea binară pe TV.

Exemple :
PLOT 25, 30
PRINT POINT (25, 30)
sau
IF POINT (x, y)=0 THEN PLOT x, y

SCREEN $

Funcţia are ca argumente poziţia alfanumerică pe ecranul TV (0<=x<=23 şi 0<=y<=31) şi dă caracterul ASCII aflat la acea poziţie video invers sau normal. Dacă nu recunoaşte caracterul dă şirul vid.

USR x

Funcţia cheamă în execuţie rutina în cod maşină de la adresa x. După ce se execută „return" rezultatul funcţiei este conţinutul registrului pereche BC.

Dacă vrem să introducem rutine în cod maşină de exemplu la adresa 32500 curăţăm memoria cu CLEAR 32499, iar de la 32500 la 32600 este un spaţiu de 100 de octeţi în care vom introduce rutina în limbaj de asamblare, cu un program BASIC de forma :

```
10 LET i=32500 : LET j=0
20 INPUT "PUNE COD MASINA :",C
30 IF C=0 THEN GO TO 70
40 POKE i,C
50 LET i=i+1 : LET j=j+1
60 GO TO 20
70 PRINT "OCTETI INTRODUSI =" ; j
```

Se salvează rutina pe casetă în fişierul BINAR cu SAVE "BINAR" CODE 32500, j unde j are valoarea printată de program.

Pentru rularea rutinei se face
RANDOMIZE USR 32500
sau : se face un program în BASIC

```
10 LOAD "" CODE 32500, J
20 RANDOMIZE USR 32500
```

care se salvează cu SAVE "PRO" LINE 10, apoi se salvează binarul cu SAVE "nume" CODE 32500, j.

La încărcarea cu LOAD "PRO", programul din 2 instrucţiuni de mai sus se lansează automat în execuţie, iar după încărcarea binarului (în linia 10) îl lansează în execuţie (linia 20).

USR x $

Funcţia are ca argument un caracter grafic definit iar ca rezultat adresa în memorie a formei binare corespunzătoare caracterului grafic definit de utilizator (începînd cu colţul stînga sus) a imaginii binare 8x8.

XVIII. VARIABILELE SISTEMULUI

Locaţiile de memorie de la 23552 la 23733 sînt utilizate special de către sistem. Prin aflarea valorii lor (cu funcţia PEEK) se pot afla informaţii despre starea sistemului.

În prima coloană din tabelul de mai jos s-au folosit următoarele notaţii :

X — asupra acestor variabile nu se va lucra cu instrucţiunile POKE întrucît s-ar putea ca sistemul să cadă.

N — modificarea valorii acestor variabile nu va avea un efect fatal.

Numărul din prima coloană este numărul de octeţi cel mai puţin semnificativ. Astfel pentru a memora o valoare "v" într-o variabilă de 2 octeţi la adresa "n" se va folosi :

POKE n,v—256 * INT (v/256)
POKE n+1, INT (v/256)
iar pentru aflarea valorii variabilei se va folosi :
PEEK n+256 * PEEK (n+1)

Numele variabilelor sînt mnemonice pentru o înţelegere mai uşoară de către utilizator a semnificaţiei variabilelor, însă ele nu vor fi referite în programe deoarece sistemul nu le recunoaşte.

NOTA	ADRESA	NUME	CONŢINUT
N8	23552	KSTATE	Utilizată la citirea tastaturii.
N1	23560	LASTK	Memorează noua tastă apăsată.
1	23561	REPDEL	Timpul (în 1/50 sec.) de repetiţie cînd o tastă este apăsată. Este iniţializată la valoarea 35.
1	23562	REPPER	Timpul între 2 repetiţii succesive cînd o tastă este ţinută apăsat (în 1/50 sec.). Iniţial are valoarea 5.
N2	23563	DEFADD	Adresa argumentelor funcţiilor definite de utilizator, dacă a fost evaluată una ; altfel este 0.
N1	23565	KDATA	Memorează culoarea tastată.
N2	23566	TVDATA	Memorează octeţi de culoare şi atributele AT şi TAB spre TV.
X38	23568	STRMS	Adresele canalelor ataşate şirurilor.
2	23606	CHARS	Adresa setului de caractere (care începe cu spaţiu şi continuă pînă la simbolul COPY) minus 256. Normal este în ROM dar se poate stabili în RAM şi să se adreseze prin CHARS.
1	23608	RASP	Lungimea semnalului de avertizare.
1	23609	PIP	Lungimea clicului pentru tastatură.
1	23610	ERRNR	Codul de eroare minus 1. Porneşte de la 255 (pentru —1).
X1	23611	FLAGS	Diferiţi indicatori pentru controlul sistemului BASIC.
X1	23612	TVFLAGS	Indicatorii asociaţi televizorului.
X2	23613	ERRSP	Adresa octetului din stiva maşină utilizat ca reîntoarcere din listarea automată.
N1	23617	MODE	Specifică cursorul K, L, C, E sau G.
2	23618	NEWPPC	Linia la care se face salt.
1	23620	NSPPC	Numărul instrucţiuni din linia la care se face salt. Cu POKE NEWPPC şi apoi POKE NSPPC face salt la o anumită instrucţiune dintr-o linie.
2	23621	PPC	Numărul liniei instrucţiunii curente în execuţie.
1	23623	SUBPPC	Numărul instrucţiunii din linie, care se execută.
1	23624	BORDER	Culoarea pentru BORDER ; conţine şi atributele ultilizate în mod normal pentru jumătatea de jos a ecranului.
2	23625	EPPC	Numărul liniei curente (unde este cursorul program).
N2	23627	VARS	Adresa variabilelor
N2	23629	DEST	Adresa variabilei ultimei asignate.
X2	23631	CHANS	Adresa canalului de date.
X2	23633	CURCHL	Adresa informaţiei curente utilizată pentru intrare şi ieşire.
X2	23635	PROG	Adresa programului BASIC.
X2	23637	NXTLIN	Adresa următoarei linii din program.
X2	23639	ELINE	Adresa marcatorului de sfîrşit a ultimului articol din DATA.
X2	23641	DATADD	Adresa ultimei comenzi introduse.
2	23643	KCUR	Adresa cursorului.
X2	23645	CHADD	Adresa următorului caracter ce trebuie interpretat ; caracterul după argumentul funcţiei PEEK sau NEW LINE la sfîrşitul unei instrucţiuni POKE.
2	23647	XPTR	Adresa caracterului de după semnul ?
X2	23649	WORKSP	Adresa spaţiului temporar de lucru.
X2	23651	STKBOT	Adresa începutului stivei calculator.
X2	23653	STKEND	Adresa de start a spaţiului liber.
N1	23655	BREG	Registrul B al calculatorului.
N2	23656	MEM	Adresa zonei utilizate pentru memoria calculator. (De obicei este MEMBOT dar nu întotdeauna).
1	23658	FLAGS2	Indicatori.
X1	23659	DFSZ	Numărul de linii (incluzînd o linie blank în partea de jos a ecranului).
2	23660	STOP	Numărul primei linii de program la listarea automată.
2	23662	OLDPPC	Numărul liniei la care sare CONTINUE.
1	23664	OSPCC	Numărul instrucţiunii din linie la care sare CONTINUE.
N1	23655	FLAGX	Diferiţi indicatori.
N2	23666	STRLEN	Adresa şirului de caractere, ultimul asignat.
N2	23668	TADDR	Adresa următorului articol din adresa de sintaxă.
2	23672	SEED	Start-ul pentru RND. Aceasta este variabila pusă de RANDOMIZE.
3	23672	FRAMES	3 octeţi (cel mai puţin semnificativ primul) numărător de cadre incrementat la fiecare 20 ms.
2	23675	UDG	Adresa primului caracter definit de utilizator. Se poate schimba salvînd spaţiu lăsînd mai puţin loc pentru caracterele grafice definite de utilizator.
1	23677	COORDS	Coordnata x a ultimului punct plotat.
1	23678		Coordonata y a ultimului punct plotat.

NOTA	ADRESA	NUME	CONȚINUT
1	23679	PPOSN	Numărul coloanei a 33-a pentru poziția printer-ului.
1	23680	PRCC	Cel mai puțin semnificativ octet al adresei noii poziții pentru LPRINT (în buffer-ul de printare).
1	23681		Nefolosită.
2	23682	ECHOE	Numărul coloanei 33 și al liniei 24 (în jumătatea de jos) pentru sfîrșitul buffer-ului de intrare.
2	23684	DFCC	Adresa în fișierul display a poziției pentru PRINT.
2	23686	DEFCCL	Ca și DFCC pentru partea de jos a ecranului.
X1	23688	SPOSN	Numărul coloanei 33 pentru poziția de PRINT.
X1	23689		Numărul liniei 24 pentru poziția de PRINT.
X2	23690	SPOSNL	Ca și la SPOSN pentru partea de jos.
1	23692	SCRCT	Numără scroll-urile ; este întotdeauna cu 1 mai mare decît numărul de scroll-uri care vor fi efectuate pînă la oprirea cu "scroll .....". Dacă la această adresă se află un număr mai mare decît 1 (să spunem 255) atunci se va efectua scroll continuu fără tipărirea mesajului de întrerupere.
1	23693	ATTRP	Culorile curente permanente, (așa cum sînt puse de instrucțiunile de culoare).
1	23694	MASKP	Utilizată pentru culorile transparente. Orice bit care este 1 arată că bitul corespunzător atributului nu este luat din ATTRP ci din ceea ce este deja pe ecran.
N1	23695	ATTRT	Culorile curente temporare, (așa cum sînt puse de articolele de culoare).
N1	23696	MASKT	Ca și MASKP dar temporar.
1	23697	PFLAG	Indicatori.
N30	23698	NEMBOT	Zona de memorie a calculatorului ; utilizată pentru memorarea numerelor care nu pot fi puse convenabil în stiva calculator.
2	23728		Neutilizate.
2	23730	RAMTOP	Adresa ultimului octet din zona BASIC.
2	23732	PRAMT	Adresa ultimului octet de RAM fizic.

XIX. MESAJE DE EROARE

Mesajele de eroare apar pe ultimele 2 linii ale ecranului (22, 23) și conțin informații despre motivul opririi. Este afișat codul erorii format dintr-o literă sau cifră, un scurt mesaj explicativ, numărul liniei BASIC și al instrucțiunii din linie unde a apărut eroare.

Forma textului de eroare este :

Cod-Mesaj scurt-număr linie : număr instrucțiune.

Mesajele de eroare se explică astfel :

0 OK

Apare în orice situație, după îndeplinirea cu succes a acțiunii, sau la un salt la o linie cu număr mai mare decît orice număr existent în programul BASIC.

1 NEXT whithout FOR

Există NEXT i dar nu există FOR i= . . . , însă există o variabilă simplă i.

2 Variabile not found

Pentru variabile simple apare cînd variabila este folosită înainte de atribuire cu LET, READ sau INPUT sau de încărcare de pe casetă, sau de atribuire în FOR-NEXT. Pentru o variabilă tablou mesajul apare cînd variabila este folosită înainte de dimensionarea cu DIM sau înainte de a fi încărcat de pe casetă.

3 Subscript wrong

Apare la variabile tablou sau la subșiruri cînd indicii depășesc dimensiunea tabloului sau șirului. Dacă indicii sînt negativi sau >65535 apare eroare B.

4 Out of memory

Apare la evaluarea expresiilor cît și la instrucțiunile LET, INPUT, FOR, DIM, GOSUB, LOAD, MERGE atunci cînd nu este destul loc în memorie. Dacă se pare că s-a blocat calculatorul, trebuie ștearsă cu DELETE linia de comandă, apoi se șterg 1—2 linii de program (care apoi se pun la loc) pentru a avea spațiu de manevră (pentru a face eventual un CLEAR).

5 Out of screen

Apare la INPUT cînd se încearcă generarea a mai mult de 23 de linii în josul ecranului și la PRINT AT cînd se printează în liniile 22 și 23.

6 Number too big

Apare în expresii aritmetice cînd calculele duc la un număr mai mare ca aproximativ 10 la puterea 38.

7 Return whithout GOSUB

Apare cînd se execută mai multe instrucțiuni RETURN decît GOSUB.

8 End of file

Apare în operații cu aparatură exterioară.

9 Stop statement

Apare cînd se execută instrucțiunea STOP în program. Tastînd CONTINUE se va continua execuția cu următoarea instrucțiune după STOP.

A Invalid argument

Apare la : SQR, LN, ASN, ACS, USR (cu argument șir) cînd argumentul este bun.

B Integer out of range

Apare la RUN, RANDOMIZE, POKE, DIM, GOTO, GOSUB, LIST, LLIST, PAUSE, PLOT,

CHR$, PEEK, USR (cu argument numeric), cînd fiind necesar un întreg, se rotunjeşte argumentul din virgula flotantă la întregul imediat mai mic, iar acest întreg este în afara domeniului permis.

C Nonsense în BASIC

Apare la VAL şi la VAL$ cînd textul din care este format argumentul nu formează o expresie validă.

D BREAK-CONT repeats

Apare la : LOAD, SAVE, VERIFY, MERGE, LPRINT, LLIST, COPY şi atunci cînd se afişează de către calculator "scroll ?" şi se tastează de utilizator N,SPACE sau STOP.

Cauza este tastarea lui BREAK în cursul operaţiilor cu periferice. Lansînd CONTINUE va fi repetată ultima instrucţiune BASIC executată. Vezi mesaj L.

E Out of DATA

Apare la READ cînd se încearcă citirea mai multor date decît existente în DATA.

F Invalid file name

Apare la SAVE cînd numele fişierului este şirul nul sau conţine mai mult de 10 caractere.

G No room for line

Apare la încercarea de introducere a unei linii în program (după ENTER) atunci cînd nu mai este spaţiu suficient în memorie pentru linie.

H STOP în INPUT

Apare la INPUT prin apariţia sau tastarea în datele de intrare a lui STOP. Reluarea cu CONTINUE repetă instrucţiunea INPUT.

I FOR whithout NEXT

Apare cînd bucla FOR nu se închide cu NEXT.

J Invalid I/O device

Apare în operaţii de cuplare cu periferice.

K Invalid colour

Apare în INVERSE şi OVER cînd numărul argument nu este potrivit (0 sau 1).

L BREAK into program

Apare la o tastare BREAK detectată între 2 linii de program. Numărul liniei şi instrucţiunea din mesaj se referă la ultima instrucţiune executată. Apăsarea tastei CONTINUE reia execuţia cu următoarea instrucţiune BASIC.

M RAMTOP no good

Apare la CLEAR (şi posibil la RUN) cînd numărul specificat variabilei de sistem RAMTOP (adresa ultimului octet de arie BASIC) este prea mare sau prea mic.

N Statement lost

Apare la RETURN, NEXT şi CONTINUE cînd se face un salt la o instrucţiune care nu există.

O invalid stream

Apare la operaţii cu periferice.

P FN whithout DEF

Apare la funcţiile definite de utilizator.

Q Parameter error

Apare la FN cînd numărul sau tipul parametrilor la apelare diferă de cei din definiţia funcţiei.

R Tape loading error

Apare la VERIFY, LOAD sau MERGE cînd a fost găsit un fişier pe casetă dar din diverse motive (demagnetizări, înregistrări cu nivel necorespunzător) nu poate fi citit sau verificat.

XX. MEMORIA RAM

La calculatorul TIM-S în primii 16K de memorie RAM, adresele 0 la 3FFFH se află interpretorul BASIC, iar ceilalţi 48K de la 4000H (16.384) la FFFF H (65.535) sînt disponibili pentru sistem şi pentru utilizator.

Împărţirea pe zone a memoriei RAM este prezentată în fig. 15.1. Zonele sînt suficient de mari pentru informaţia pe care o conţin şi dacă se introduce mai mult într-un loc totul deasupra acelui loc se deplasează în sus şi invers ; se deplasează în jos dacă se şterg informaţii.

Atributele de culoare sînt memorate în ordine linie după linie.

Buffer-ul de printare memorează caracterele destinate imprimantei.

Variabilele de sistem conţin informaţii despre starea în care este calculatorul ; sînt listate în capitolul anterior.

Ele nu sînt variabile BASIC şi numele lor nu vor fi recunoscute de calculator.

Spaţiul destinat Microdriv-ului este utilizat numai cu Microdriver ; normal în acest spaţiu nu este memorat nimic.

Spaţiul pentru informaţiile de canal conţine informaţii despre echipamentele de intrare şi ieşire (tastatură, ecran, imprimantă).

Fiecare linie de program BASIC are forma prezentată în fig. 15.2. Spre deosebire de toate celelalte cazuri de numere pe 2 octeţi, numărul liniei este memorat cu octelul cel mai semnificativ primul, deci în ordinea în care se introduce numărul.

O constantă numerică în program este reprezentată utilizînd caracterul CHR$ 14 urmat de valoarea constantei reprezentată pe 5 octeţi.

Variabilele au diferite forme de reprezentare în funcţie de natura lor.

Exemplificarea este făcută în fig. 15.3. şi 15.4. Pentru tablouri vezi fig. 15.5 ; ordinea elementelor este :

primele : elemente pentru care primul indice este 1

apoi : elemente pentru care primul indice este 2

apoi : elemente pentru care primul indice este 3

ş.a.m.d. pentru toate valorile posibile ale primului indice.

Elementele pentru primul indice dat sînt ordonate în acelaşi mod utilizînd al doilea indice etc. pînă la ultimul indice.

Variabila de control pentru o buclă FOR-NEXT se reprezintă ca în fig. 15.6. Pentru şiruri

de caractere şi tablouri de şiruri de caractere reprezentarea este prezentată în fig. 15.7 respectiv 15.8. Stiva calculator conţine numerele asupra cărora se operează în timpul execuţiei programului.

Stiva maşină este stiva utilizată de Z-80 pentru reţinerea adreselor de întoarcere etc.

Stiva GOSUB conţine numărul de linie şi numărul de instrucţiune al instrucţiunii GOSUB (aceasta constituie adresa de întoarcere) astfel încît după execuţia subrutinei, la întîlnirea instrucţiunii RETURN calculatorul ia această adresă din stiva GOSUB şi execută prima instrucţiune după ea.

RAMTOP reprezintă cea mai mare adresă utilizată de sistemul BASIC. Fiecare comandă NEW care curăţă RAM-ul face acest lucru numai pînă la adresa RAMTOP, deci nu modifică spaţiul destinat caracterelor grafice definite de utilizator. Adresa RAMTOP se poate schimba cu instrucţiunea CLEAR :

CLEAR (nou RAMTOP)

Aceasta face următoarele :

— şterge toate variabilele

— curăţă fişierul DISPLAY (ca şi CLS)

— setează poziţia de PLOT în colţul din stînga jos

— execută RESTORE

— curăţă stiva GOSUB şi o pune la noul RAMTOP — presupunînd că noul RAMTOP este cuprins între stiva calculator şi ultima adresă de RAM fizic — altfel adresa de RAMTOP este lăsată neschimbată.

RUN face de asemenea CLEAR dar nu schimbă adresa RAMTOP.

În acest fel cu instrucţiunea CLEAR, mutînd adresa de RAMTOP în sus se eliberează mai mult loc pentru BASIC prin scriere peste spaţiul destinat caracterelor grafice definite de utilizator, sau se poate muta RAMTOP în jos mărind astfel spaţiul asupra căruia acţionează comanda NEW.

Orice număr (cu excepţia lui 0) poate fi scris ca + sau — m*(∆e) unde :

— + sau — este semnul

— m este mantisa cuprinsă între 1/2 şi 1 (exclusiv 1)

— e este exponentul, un număr întreg (poate fi şi negativ).

Dacă vrem să reprezentăm în binar trebuie să ţinem cont că este un număr fracţionar, deci va avea un punct binar (similar cu punctul zecimal în baza 10) şi apoi partea fracţionară ; deci în binar :

o jumătate este —.1

un sfert este —.01

trei sferturi sînt —.11 ş.a.m.d.

Întrucît m este mai mic decît 1 înseamnă că are biţi înainte de punctul binar şi deoarece este mai mare decît 1/2 primul bit după punctul binar este 1.

Pentru memorarea numerelor în calculator se utilizează 5 octeţi după care urmează :

1. Se scriu primii 8 biţi ai mantisei în octetul al doilea (ştim că primul bit este 1), următorii 8 biţi în octetul al treilea, următorii 8 biţi în octetul 5.

2. Se înlocuieşte primul bit din al doilea octet (care este 1) cu semnul : 0 pentru plus şi 1 pentru minus.

3. Se scrie (exponentul +128) în primul octet.

De exemplu presupunem că avem numărul 1/10 ; atunci avem :

1/10=4/5 * 2 ∧ —3

Astfel mantisa m în binar este :

11001100110011001100110011001100...

Întrucît al 33-lea bit este 1 se rotunjeşte bitul 32 din 0 în 1.

Exponentul este —3 şi aplicînd regulile de mai sus se obţin următorii 5 octeţi :

01111101 01001100 11001100 11001100 11001101

—3+128 mantisa

Se observă că primul bit al mantisei (octetul al doilea) este 0 pentru semnul pozitiv al numărului.

Pentru memorarea numerelor întregi între —65535 şi +65535 există o altă metodă :

1. primul octet este 0

2. al doilea octet este 0 pentru un număr pozitiv sau FF pentru un număr negativ

3. al treilea şi al patrulea octet sînt cel mai puţin semnificativ octet şi respectiv cel mai semnificativ octet ai numărului ce trebuie reprezentat (sau ai numărului de reprezentat +131072 dacă acesta este negativ)

4. al cincilea octet este 0.

Fig. 15.1. Organizarea memoriei RAM

0 ... 16383	: Interpretorul BASIC
16384 ... 22527	: Fişierul imaginii de pe ecran
22528 ... 23295	: Spaţiul cu atribute ale ecranului
23296 ... 23551	: Fişierul caracterelor de tipărit
23552 ... 23733	: Variabile de sistem
23734 ... CHANS	: Spaţiu utilizat de microdriver
CHANS ... PROG-1	: Informaţii despre canale (+ # 80)
PROG ... VARS-1	: Textul programului BASIC
VARS ... ELINE-1	: Variabilele programului (+ # 80)
ELINE ... WORKSP-1	: Spaţiu utilizat la editare (+ # OD80)
WORKSP ... STKBOT-1	: Spaţiu utilizat la introducere date (la INPUT) şi ca spaţiu temporar de lucru (+ # OD)
STKBOT ... STKEND	: Stiva folosită la calcule aritmetice
STKEND ... SP-1	: Spaţiu liber
SP ... RAMTOP-1	: Stiva folosită de microprocesor urmată de stiva folosită de instrucţiunea GOSUB

RAMTOP : Ultima adresă disponibilă interpretorului BASIC (conține # 3E)

UDG ... PRAMT : Fişier ce conţine caractere grafice definite de utilizator

(NOTĂ : 1. În harta memoriei apar variabile de sistem ; numele lor nu este recunoscut de calculator.

2. Semnul # este scris înaintea cifrelor hexazecimale).

Fig. 15.2. Reprezentarea unei linii BASIC

1. Numărul unei linii BASIC : 2 octeţi
2. Lungimea textului liniei (inclusiv ENTER : 2 octeţi
3. Textul liniei BASIC : x octeţi
4. ENTER (# OD) : 1 octet

Exemplu Linia BASIC 10 GO TO 3000

1. #000A	(10)	: 2 octeţi
2. # 0C00	(12)	: 2 octeţi
3. # EC	(GOTO)	:
# 33303030	(3000)	
# 0E0000B80B00	(constanta 3000)	: 11 octeţi
4. # 0D	(ENTER)	: 1 octet

Fig. 15.3. Variabila cu nume de o literă

1. Tipul variabilei pe primii 3 biţi este (011) şi numele variabilei compactat pe 5 biţi prin scădere # 60 din codul ASCII al literei : 1 octet
2. Exponentul : 1 octet
3. Mantisa : 4 octeţi

Exemplu Variabila a atribuită cu +0.75

1. #61 sau 01100001 (a-compactat) : 1 octet
2. # 80 sau 10000000 sau 128 (exponent=0) : 1 octet
3. # 40000000 sau 0100 0000 0000 0000 0000 0000 0000 0000 din 11=0.75 şi +0.75=.01 (convenţie) : 4 octeţi

Fig. 15.4. Variabila cu nume mai lung de o literă

1. Tipul pe 3 biţi este (101) şi prima literă compactată (# 60) : 1 octet
2. Următoarele caractere din nume : x octet
3. Ultimul caracter din nume avînd bitul cel mai semnificativ pus pe 1 : 1 octet
4. exponent : 1 octet
5. mantisa : 4 octeţi

Fig. 15.5 Variabila tablou numeric

1. Tipul pe 3 biţi este (100) şi prima literă din nume compactată (# 60) : 1 octet
2. Lungimea totală a fişierului ce urmează : 2 octeţi
3. Număr de dimensiuni : 1 octet
4. Prima dimensiune ... ultima dimensiune : 2x octeţi
5. Elementele tabloului : 5 octeţi/element

Fig. 15.6 Variabila buclei FOR

1. Tipul pe 3 biţi este (111) şi prima literă din nume compactată (# 60) : 1 octet
2. Valoarea curentă : 5 octeţi
3. Valoarea limită : 5 octeţi
4. Pasul : 5 octeţi
5. Numărul liniei BASIC ce conţine instrucţiunea FOR asociată variabilei : 2 octeţi
6. Numărul instrucţiunii din linia de la 5 ce urmează instrucţiunii FOR : 1 octet

Fig. 15.7. Variabila şir de caractere

1. Tipul pe 3 biţi este (010) şi litera nume compactată (# 60) : 1 octet
2. Numărul de caractere ale şirului : 2 octeţi
3. Caracterele şirului : 1 octet/caracter

Fig. 15.8. Variabila tablou de caractere

1. Tipul pe 3 biţi este (110) şi litera nume compactată (60) : 1 octet
2. Lungimea fişierului ce urmează : 2 octeţi
3. Număr de dimensiuni : 1 octet
4. Prima dimensiune ... ultima dimensiune (x dimens.) : 2x octeţi
5. Elementele tabloului pe cîte 1 octet : 1 octet / caracter

XXI. SETUL DE CARACTERE

Se prezintă în continuare setul complet de caractere al calculatorului TIM-S, cu codurile zecimal şi hexazecimal. Dacă se interpretează codurile drept coduri de instrucţiuni maşină pentru Z-80 atunci coloanele din dreapta dau mnemonicele corespunzătoare limbajului de asamblare. Anumite instrucţiuni pentru Z-80 încep cu CB sau ED ; acestea sînt date în cele două coloane din dreapta.

NOTAŢII : n — neutilizat
g.u. — caractere grafice definite de utilizator
p.i. — prefix la instrucţiunile care utilizează ix sau iy
po — paritate impară
pe — paritate pară

COD	CARACTER	HEXA	Z80	DUPĂ CB	DUPĂ ED
0	n	00	nop	rlc b	
1	n	01	ld bc, NN	rlc c	
2	n	02	ld (bc), a	rlc d	
3	n	03	inc bc	rlc e	
4	n	04	inc b	rlc h	
5	n	05	dec b	rlc l	
6	PRINT virgulă	06	ld b, N	rlc (hl)	
7	EDIT	07	rlca	rlc a	
8	cursor stînga	09	ex af, af'	rrc b	
9	cursor dreapta	09	add hl, bc	rrc c	
10	cursor jos	0A	ld a, (bc)	rrc d	
11	cursor sus	0B	dec bc	rrc e	
12	DELETE	0C	inc c	rrc h	
13	ENTER	0D	dec c	rrc l	
14	număr	0E	ld c, N	rrc (hl)	
15	neutilizat	0F	rrca	rrc a	
16	control INK	10	djnz DIS	rl b	
17	control PAPER	11	ld de, NN	rl c	
18	control FLASH	12	ld (de), a	rl d	
19	control BRIGHT	13	inc de	rl e	
20	control INVERSE	14	inc d	rl h	
21	control OVER	15	dec d	rl l	
22	control AT	16	ld d, N	rl (hl)	
23	control TAB	17	rla	rl a	
24	n	18	jr DIS	rr b	
25	n	19	add hl, de	rr c	
26	neutilizat	1A	ld a, (de)	rr d	
27	·	1B	dec de	rr e	
28	·	1C	inc e	rr h	
29		1D	dec e	rr l	
30	n	1E	ld e, N	rr (hl)	
31	n	1F	rra	rr a	
32	spațiu	20	jr nz, DIS	sla b	
33	!	21	ld hl, NN	sla c	
34	"	22	ld (NN), hl	sla d	
35		23	inc hl	sla e	
36	$	24	inc h	sla h	
37	%	25	dec h	sla l	
38	&	26	ld h, N	sla (hl)	
39	'	27	daa	sla a	
40	(	28	jr z, DIS	sra b	
41	)	29	add hl, hl	sra c	
42	*	2A	ld hl, (NN)	sra d	
43	+	2B	dec hl	sra e	
44	,	2C	inc l	sra h	
45	—	2D	dec l	sra l	
46	.	2E	ld l, N	sra (hl)	
47	/	2F	cpl	sra a	
48	0	30	jr nc, DIS		
49	1	31	ld sp, NN		
50	2	32	ld (NN), a		
51	3	33	inc sp		
52	4	34	inc (hl)		
53	5	35	dec (hl)		
54	6	36	ld (hl), N		
55	7	37	sc f		
56	8	38	jr c, DIS	srl b	
57	9	39	add hl, sp	srl c	
58	:	3A	ld a, (NN)	srl d	
59	;	3B	dec sp	srl e	
60	<	3C	inc a	srl h	
61	=	3D	dec	srl l	
62	>	3E	ld a, N	srl (hl)	
63	?	3F	ccf	srl a	
64	C	40	ld b, b	bit 0, b	in b, (c)
65	A	41	ld b, c	bit 0, c	out (c), b
66	B	42	ld b, d	bit 0, d	sbc ul, bc
67	C	43	ld b, e	bit 0, e	ld (NN), bc
68	D	44	ld b, h	bit 0, h	neg
69	E	45	ld b, l	bit 0, l	retn
70	F	46	ld b, (hl)	bit 0, (hl)	im o
71	G	47	ld b, a	bit 0, a	ld i, a
72	H	48	ld c, b	bit 1, b	in c, (c)
73	I	49	ld c, c	bit 1, c	out (c), c
74	J	4A	ld c, d	bit 1, e	adc hl, bc
75	K	4B	ld c, e	bit 1, d	ld bc, (NN)

76	L	4C	ld c, h	bit 1, h	
77	M	4D	ld c, l	bit 1, l	reti
78	N	4E	ld c, (hl)	bit 1, (hl)	
79	O	4F	ld c, a	bit 1, a	ld r, a
80	P	50	ld d, b	bit 2, b	in d, (c)
81	Q	51	ld d, c	bit 2, c	out (c), d
82	R	52	ld d, d	bit 2, d	sbc hl, de
83	S	53	ld d, e	bit 2, e	ld (NN), de
84	T	54	ld d, h	bit 2, h	
85	U	55	ld d, l	bit 2, l	
86	V	56	ld d, (hl)	bit 2, (hl)	im 1
87	W	57	ld d, a	bit 2, a	ld a, i
88	X	58	ld e, b	bit 3, b	in e, (c)
89	Y	59	ld e, c	bit 3, c	out (c), e
90	Z	5A	ld e, d	bit 3, d	adc hl, de
91	[	5B	ld e, e	bit 3, e	ld de, (NN)
92	/	5C	ld e, h	bit 3, l	
93	]	5D	ld e, l	bit 3, h	
94		5E	ld e, (hl)	bit 3, (hl)	im 2
95	—	5F	ld e, a	bit 3, a	ld a, r
96		60	ld h, b	bit 4, b	in h, (c)
97	a	61	ld h, c	bit 4, c	out (c), h
98	b	62	ld h, d	bit 4, d	sbc hl, hl
99	c	63	ld h, e	bit 4, e	ld (NN), hl
100	d	64	ld h, h	bit 4, h	
101	e	65	ld h, l	bit 4, l	
102	f	66	ld h, (hl)	bit 4, (hl)	
103	g	67	ld h, a	bit 4, a	rrd
104	h	68	ld l, b	bit 5, b	in l, (c)
105	i	69	ld l, c	bit 5, c	out (c), l
106	j	6A	ld l, d	bit 5, d	adc hl, hl
107	k	6B	ld l, e	bit 5, e	ld hl ,(NN)
108	l	6C	ld l, h	bit 5, h	
109	m	6D	ld l, l	bit 5, l	
110	n	6E	ld l, (hl)	bit 5, (hl)	
111	o	6F	ld l, a	bit 5, a	rld
112	p	70	ld (hl), b	bit 6, b	in f, (c)
113	q	71	ld (hl), c	bit 6, c	
114	r	72	ld (hl), d	bit 6, d	
115	s	73	ld (hl), e	bit 6, e	sbc hl, sp
116	t	74	ld (hl), h	bit 6, h	ld (NN), sp
117	u	75	ld (hl), l	bit 6, l	
118	v	76	halt	bit 6, (hl)	
119	w	77	ld (hl), a	bit 6, a	
120	x	78	ld a, b	bit 7, b	
121	y	79	ld a, c	bit 7, c	in a, (c)
122	z	7A	ld a, d	bit 7, d	out (c), a
123		7B	ld a, e	bit 7, e	adc hl, sp
124	;	7C	ld a, h	bit 7, h	ld sp, (NN)
125		7D	ld a, l	bit 7, l	
126		7E	ld a, (hl)	bit 7, (hl)	
127	©	7F	ld a, a	bit 7, a	
128		80	add a, b	res 0, b	
129		81	add a, c	res 0, c	
130		82	add a, d	res 0, d	
131		83	add a, e	res 0, e	
132		84	add a, h	res 0, h	
133		85	add a, l	res 0, l	
134		86	add a, (hl)	res 0, (hl)	
135		87	add a, a	res 0, a	
136		88	adc a, b	res 1, b	
137		89	adc a, c	res 1, c	
138		8A	adc a, d	res 1, d	
139		8B	adc a, e	res 1, e	
140		8C	adc a, h	res 1, h	
141		8D	adc a, l	res 1, l	
142		8E	adc a, (hl)	res 1, (hl)	
143	(a)	8F	adc a, a	res 1, a	
144	(a) g.u.	90	sub b	res 2, b	
145	(b) g.u.	91	sub c	res 2, c	
146	(c) g.u.	92	sub d	res 2, d	
147	(d) g.u.	93	sub e	res 2, e	
148	(e) g.u.	94	sub h	res 2, h	
149	(f) g.u.	95	sub l	res 2, l	

150	(g) g.u.	96	sub (hl)	res 2, (hl)	
151	(h) g.u.	97	sub a	res 2, a	
152	(i) g.u.	98	sbc a, b	res 3, b	
153	(j) g.u.	99	sbc a, c	res 3, c	
154	(k) g.u.	9A	sbc a, d	res 3, d	
155	(l) g.u.	9B	sbc a, e	res 3, e	
156	(m) g.u.	9C	sbc a, h	res 3, h	
157	(n) g.u.	9D	sbc a, l	res 3, l	
158	(o) g.u.	9E	sbc a, (hl)	res 3, (hl)	
159	(p) g.u.	9F	sbc a, a	res 3, a	
160	(q) g.u.	A0	and b	res 4, b	ldi
161	(r) g.u.	A1	and c	res 4, c	cpi
162	(s) g.u.	A2	and d	res 4, d	ini
163	(t) g.u.	A3	and e	res 4, e	outi
164	(u) g.u.	A4	and h	res 4, h	
165	RND	A5	and l	res 4, l	
166	INKEY $	A6	and (hl)	res 4, (hl)	
167	PI	A7	and a	res 4, a	
168	FN	A8	xor b	res 5, b	ldd
169	POINT	A9	xor c	res 5, c	cpd
170	SCREEN $	AA	xor d	res 5, d	ind
171	ATTR	AB	xor e	res 5, e	outd
172	AT	AC	xor h	res 5, h	
173	TAB	AD	xor l	res 5, l	
174	VAL$	AE	xor (hl)	res 5, (hl)	
175	CODE	AF	xor a	res 5, a	
176	VAL	B0	or b	res 6, b	ldir
177	LEN	B1	or c	res 6, c	cpir
178	SIN	B2	or d	res 6, d	inir
179	COS	B3	or e	res 6, e	otir
180	TAN	B4	or h	res 6, h	
181	ASN	B5	or l	res 6, l	
182	ACS	B6	or (hl)	res 6, (hl)	
183	ATN	B7	or a	res 6, a	
184	LN	B8	cp b	res 7, b	lddr
185	EXP	B9	cp c	res 7, c	cpdr
186	INT	BA	cp d	res 7, d	indr
187	SQR	BB	cp e	res 7, e	otdr
188	SGN	BC	cp h	res 7, h	
189	ABS	BD	cp l	res 7, l	
190	PEEK	BE	cp (hl)	res 7, (hl)	
191	IN	BF	cp a	res 7, a	
192	USR	C0	ret nz	set 0, b	
193	STR $	C1	pop bc	set 0, c	
194	CHR $	C2	jp nz, NN	set 0, d	
195	NOT	C3	jp NN	set 0, e	
196	BIN	C4	call nz, NN	set 0, h	
197	OR	C5	push bc	set 0, l	
198	AND	C6	add a, N	set 0, (hl)	
199	<=	C7	rst O	set 0, a	
200	>=	C8	ret z	set 1, b	
201	<>	C9	ret	set 1, c	
202	LINE	CA	jp z, NN	set 1, d	
203	THEN	CB		set 1, e	
204	TO	CC	call z, NN	set 1, h	
205	STEP	CD	call NN	set 1, l	
206	DEF FN	CE	adc a, N	set 1, (hl)	
207	CAT	CF	rst 8	set 1, a	
208	FORMAT	D0	ret nc	set 2, b	
209	MOVE	D1	pop de	set 2, c	
210	ERASE	D2	jp nc, NN	set 2, d	
211	OPEN $	D3	out (N), a	set 2, e	
212	CLOSE $	D4	call hc, NN	set 2, h	
213	MERGE	D5	push de	set 2, l	
214	VERIFY	D6	sub N	set 2, (hl)	
215	BEEP	D7	rst 16	set 2, a	
216	CIRCLE	D8	ret c	set 3, b	
217	INK	D9	exx	set 3, c	
218	PAPER	DA	jp c, NN	set 3, d	
219	FLASH	DB	in a, (N)	set 3, e	
220	BRIGHT	DC	call c, NN	set 3, h	
221	INVERSE	DD	p.i. (ix)	set 3, l	
222	OVER	DE	sbc a, N	set 3, (hl)	
223	OUT	DF	rst 24	set 3, a	
224	LPRINT	E0	ret po	set 4, b	
225	LLIST	E1	pop hl	set 4, c	
226	GOTO	E2	jp po, NN	set 4, d	
227	READ	E3	ex (sp), hl	set 4, e	
228	DATA	E4	call po, NN	set 4, h	
229	RESTORE	E5	push hl	set 4, l	
230	NEW	E6	and N	set 4, (hl)	
231	BORDER	E7	rst 32	set 4, a	

232	CONTINUE	E8	ret pe	set 5, b
233	DIM	E9	jp (hl)	set 5, c
234	REM	EA	jp pe, NN	set 5, d
235	FOR	EB	ex de, hl	set 5, e
236	GOTO	EB	call pe, NN	set 5, h
237	GOSUB	ED		set 5, l
238	INPUT	EE	xor N	set 5, (hl)
239	LOAD	EF	rst 40	set 5, a
240	LIST	F0	ret p	set 6, b
241	LET	F1	pop af	set 6, c
242	PAUSE	F2	jp p, NN	set 6, d
243	NEXT	F3	di	set 6, e
244	POKE	F4	call p, NN	set 6, h
245	PRINT	F5	push af	set 6, l
246	PLOT	F6	or N	set 6, (hl)
247	RUN	F7	rst 48	set 6, a
248	SAVE	F8	ret m	set 7, b
249	RANDOMIZE	F9	ld sp, hl	set 7, c
250	IF	FA	jp m, NN	set 7, d
251	CLS	FB	ei	set 7, e
252	DRAW	FC	call m, NN	set 7, h
253	CLEAR	FD	p.i. (iy)	set 7, l
254	RETURN	FE	cp N	set 7, (hl)
255	COPY	FF	rst 56	set 7, a

XXII. INDEX

În index se prezintă modul de obţinere de la tastatură a funcţiilor instrucţiunilor şi simbolurilor ASCII.

NOTAŢII :

/K/, /L/, /C/, /G/, /E/ = modurile de lucru (instrucţiune, litere mici, litere mari, grafic, extins).

CS+X=tasta CAPS SHIFT apăsată simultan cu tasta X.

SS+X=tasta SYMBOL SHIFT apăsată simultan cu tasta X.

A, ..., Z=tasta înscrisă cu litera respectivă.

0, ..., 9=tasta înscrisă cu cifra respectivă.

SIMBOL	Obţinere
!	/K/, /L/, /C/, SS+1
"	/K/, /L/, /C/, SS+P
#	/K/, /L/, /C/, SS+3
$	/K/, /L/, /C/, SS+4
%	/K/, /L/, /C/, SS+5
&	/K/, /L/, /C/, SS+6
'	/K/, /L/, /C/, SS+7
(	/K/, /L/, /C/, SS+8
)	/K/, /L/, /C/, SS+9
*	/K/, /L/, /C/, SS+B
+	/K/, /L/, /C/, SS+K
,	/K/, /L/, /C/, SS+N
—	/K/, /L/, /C/, SS+J
	/K/, /L/, /C/, SS+M
/	/K/, /L/, /C/, SS+V
:	/K/, /L/, /C/, SS+Z
;	/K/, /L/, /C/, SS+O
<	/K/, /L/, /C/, SS+R
=	/K/, /L/, /C/, SS+L
>	/K/, /L/, /C/, SS+T
?	/K/, /L/, /C/, SS+C
a	/K/, /L/, /C/, SS+2
[	/E/, SS+Y
	/E/, SS+D
]	/E/, SS+U
∧	/K/, /L/, /C/, SS+H
—	/K/, /L/, /C/, SS+0
C	/K/, /L/, /C/, SS+X
}	/E/, SS+F
¦	/E/, SS+S
{	/E/, SS+G
~	/E/, SS+A
$	/E/, SS+P
<=	/K/, /L/, /C/, SS+Q
>=	/K/, /L/, /C/, SS+W

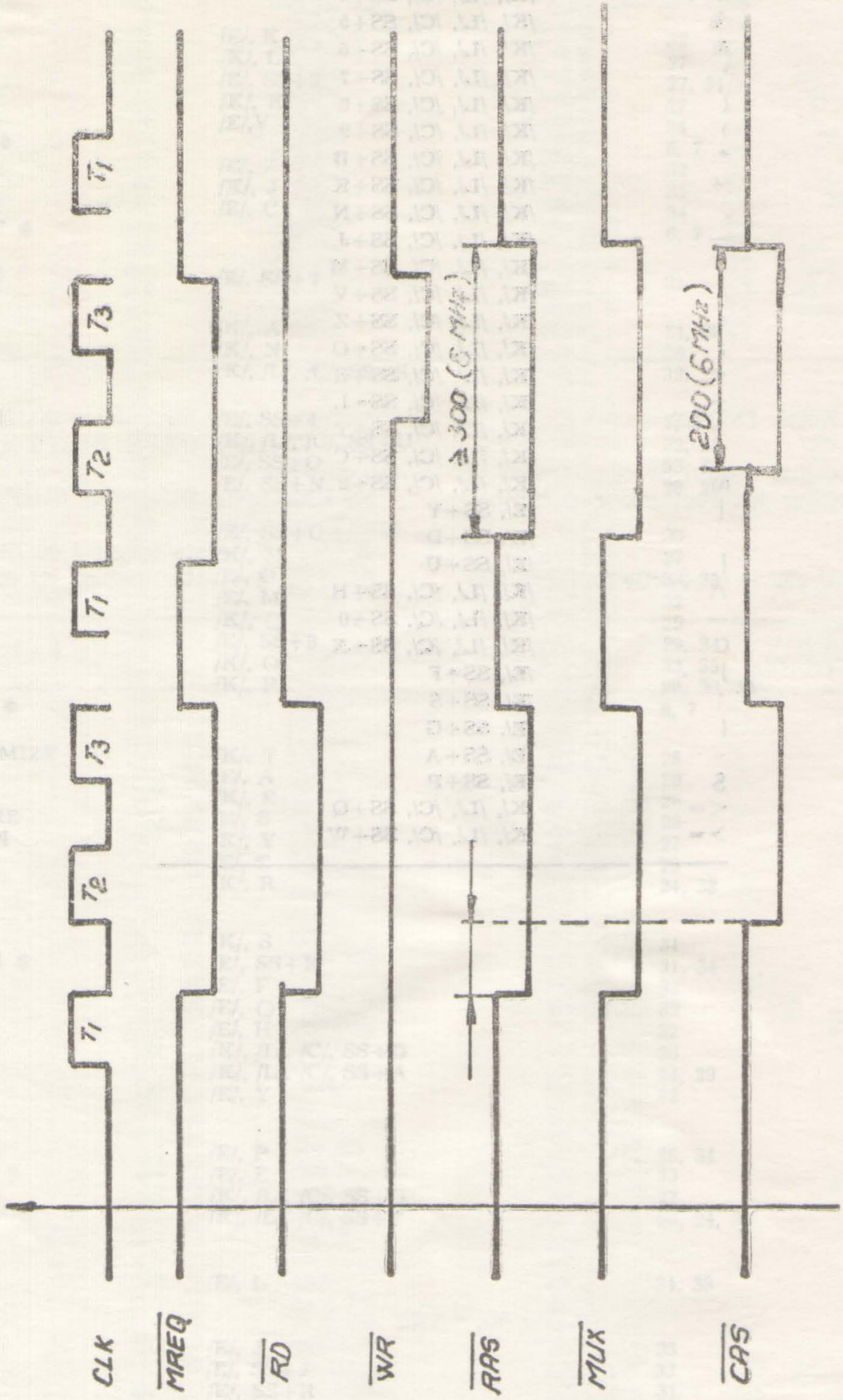


Fig. 2.1.

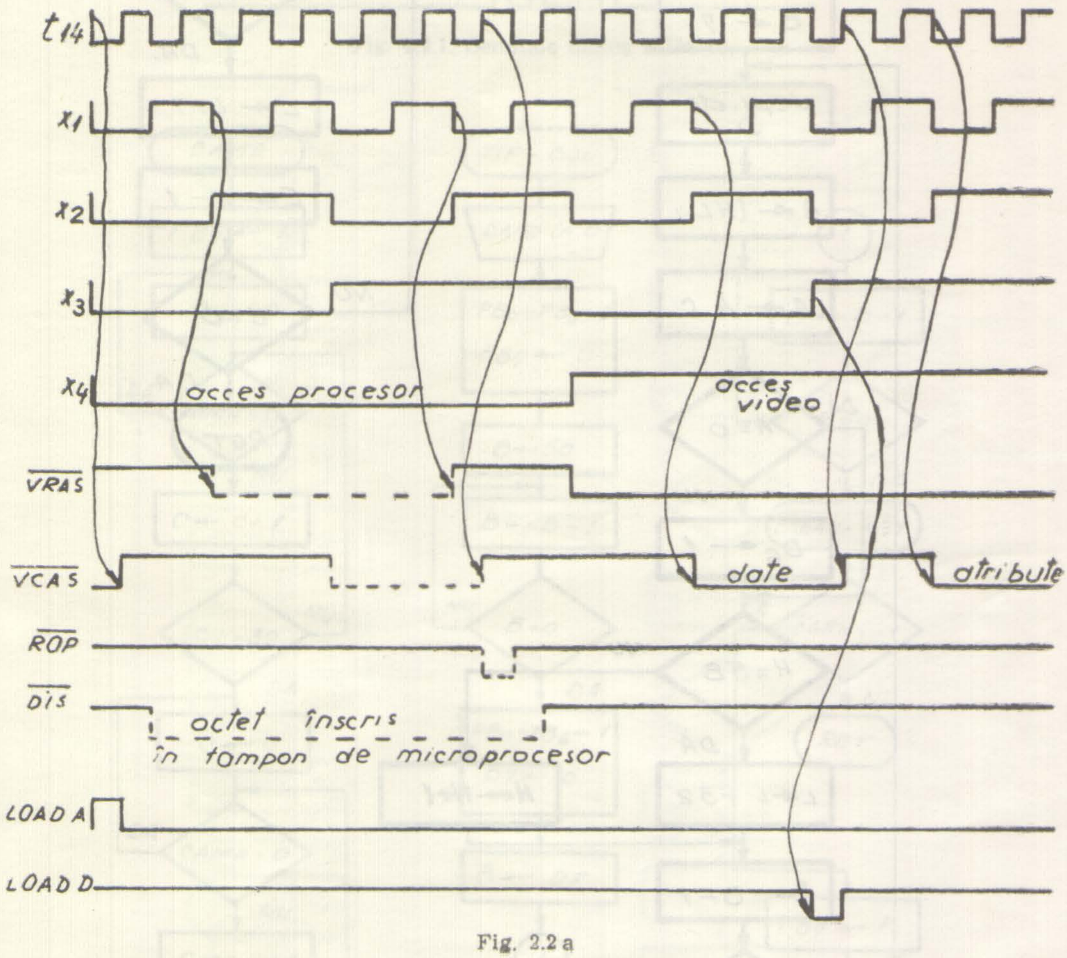

t14
x1
x2
x3
x4
acces procesor
acces video
VRAS
VCAS
date
atribute
ROP
DIS
octet înscris
în tampon de microprocesor
LOAD A
LOAD D

Fig. 2.2 a

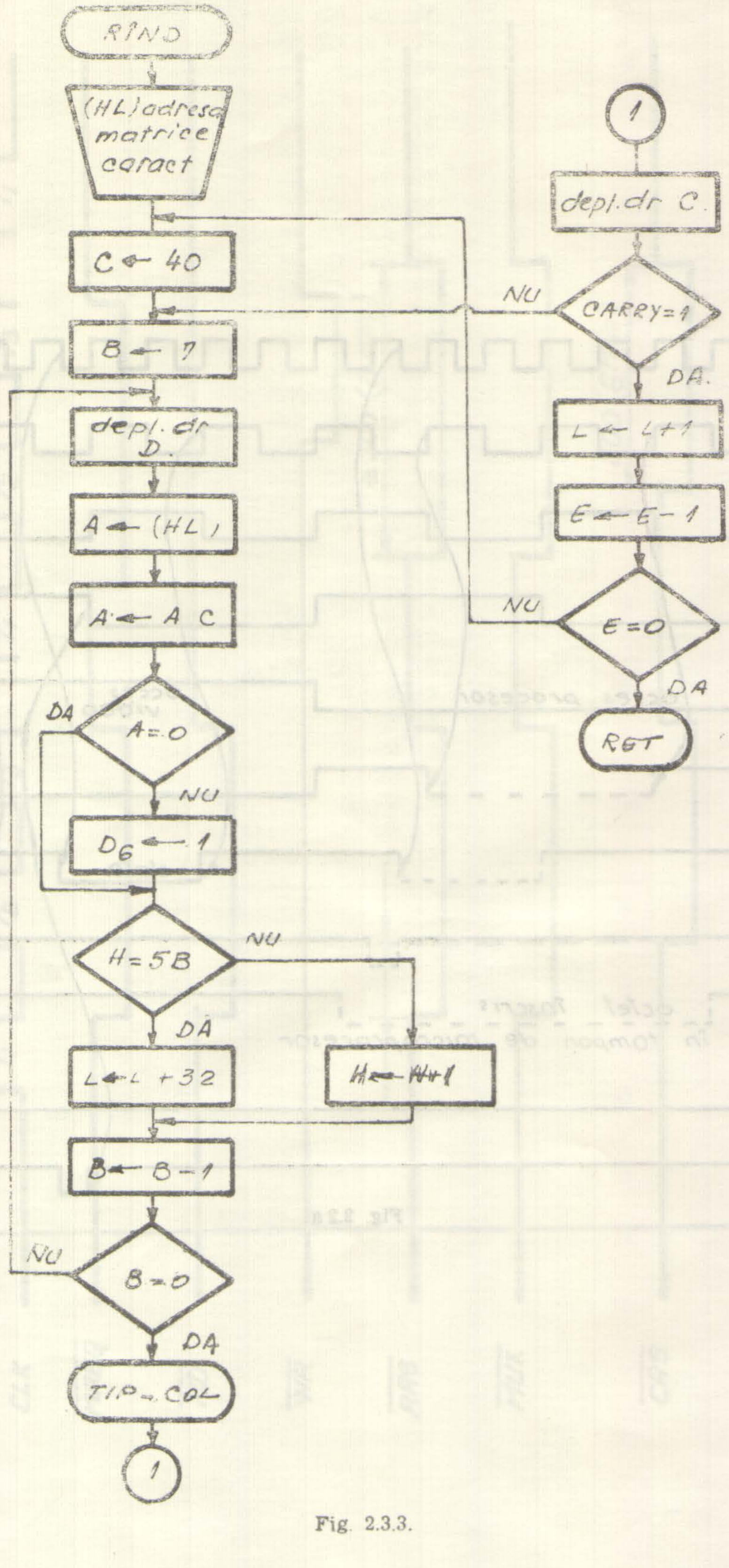

RÎND
(HL) adresa matrice caract
C ← 40
B ← 7
depl. dr D.
A ← (HL)
A ← A C
A=0
DA
NU
D6 ← 1
H=5B
NU
DA
L ← L +32
H ← H+1
B ← B-1
B=0
NU
DA
TIP_COL
1
1
depl. dr C.
CARRY=1
NU
DA.
L ← L+1
E ← E-1
E=0
NU
DA
RET

Fig. 2.3.3.

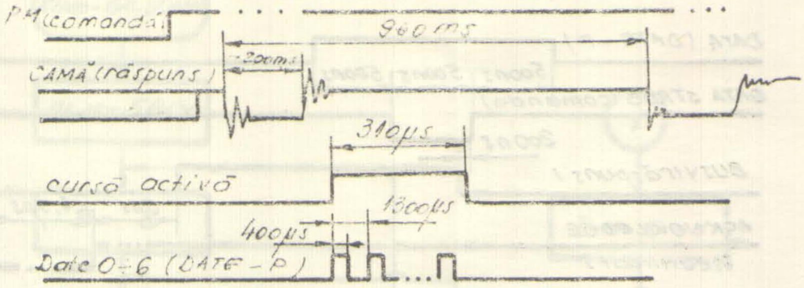


Fig. 2.2.1. Semnale dialog MIM-40.

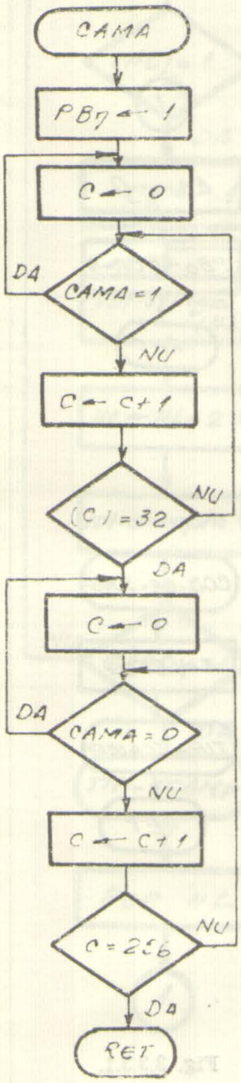


Fig. 2.3.1.

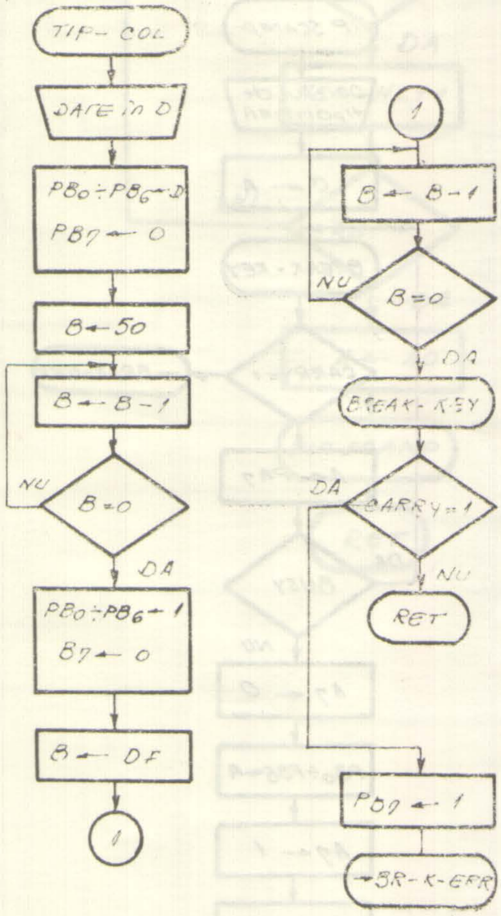


Fig. 2.3.2.

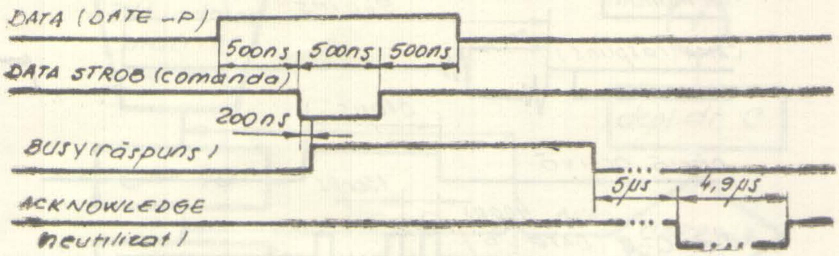


Fig. 3.2. Semnale dialog SCAMP.

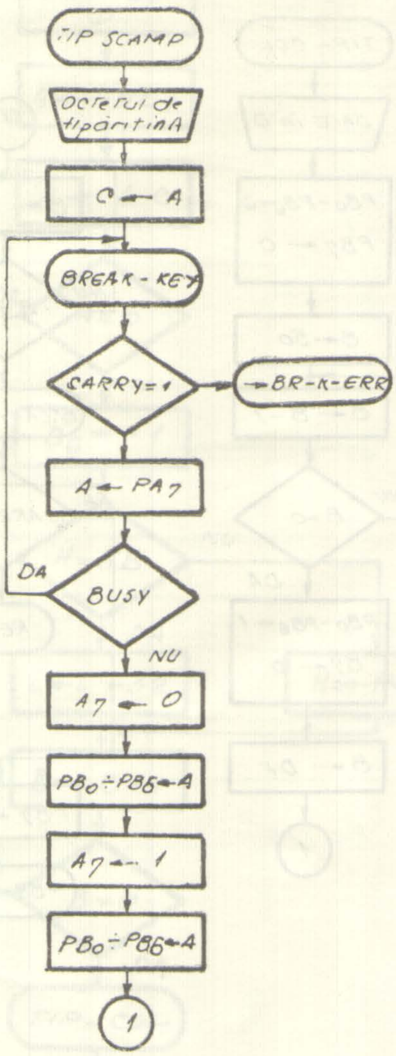


Fig. 3.3.2.

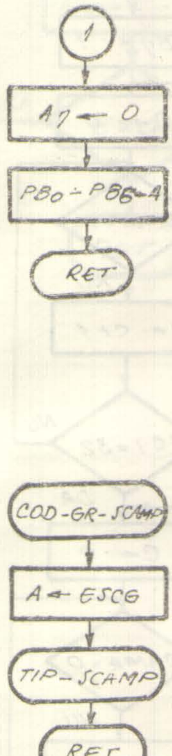


Fig. 3.3.3.1.

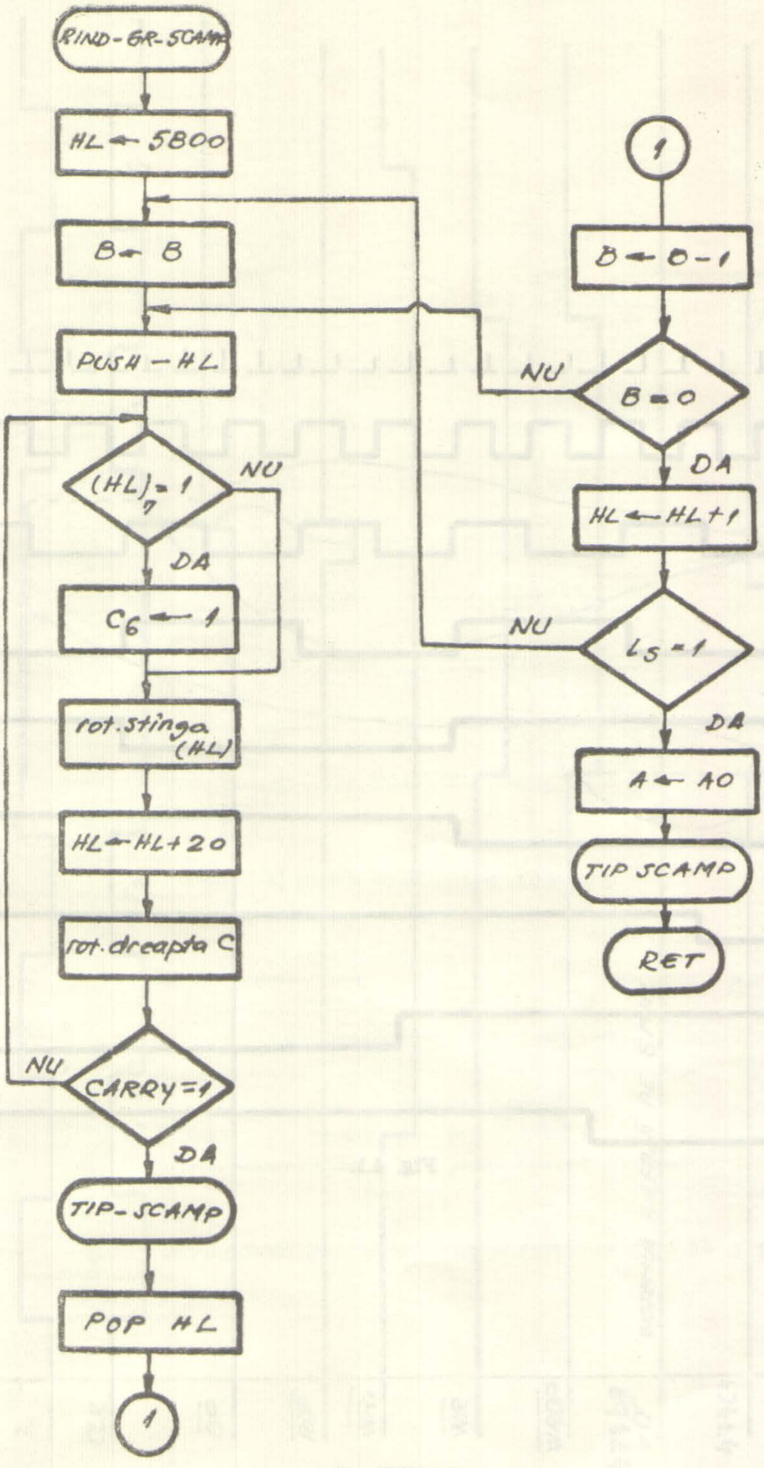

RIND-GR-SCAMP
HL ← 5800
B ← 8
PUSH-HL
(HL)7 = 1
NU
DA
C6 ← 1
rot.stinga (HL)
HL ← HL+20
rot.dreapta C
CARRY=1
NU
DA
TIP-SCAMP
POP HL
1
1
B ← B-1
B=0
NU
DA
HL ← HL+1
L5=1
NU
DA
A ← A0
TIP SCAMP
RET

Fig. 3.6.2.2.

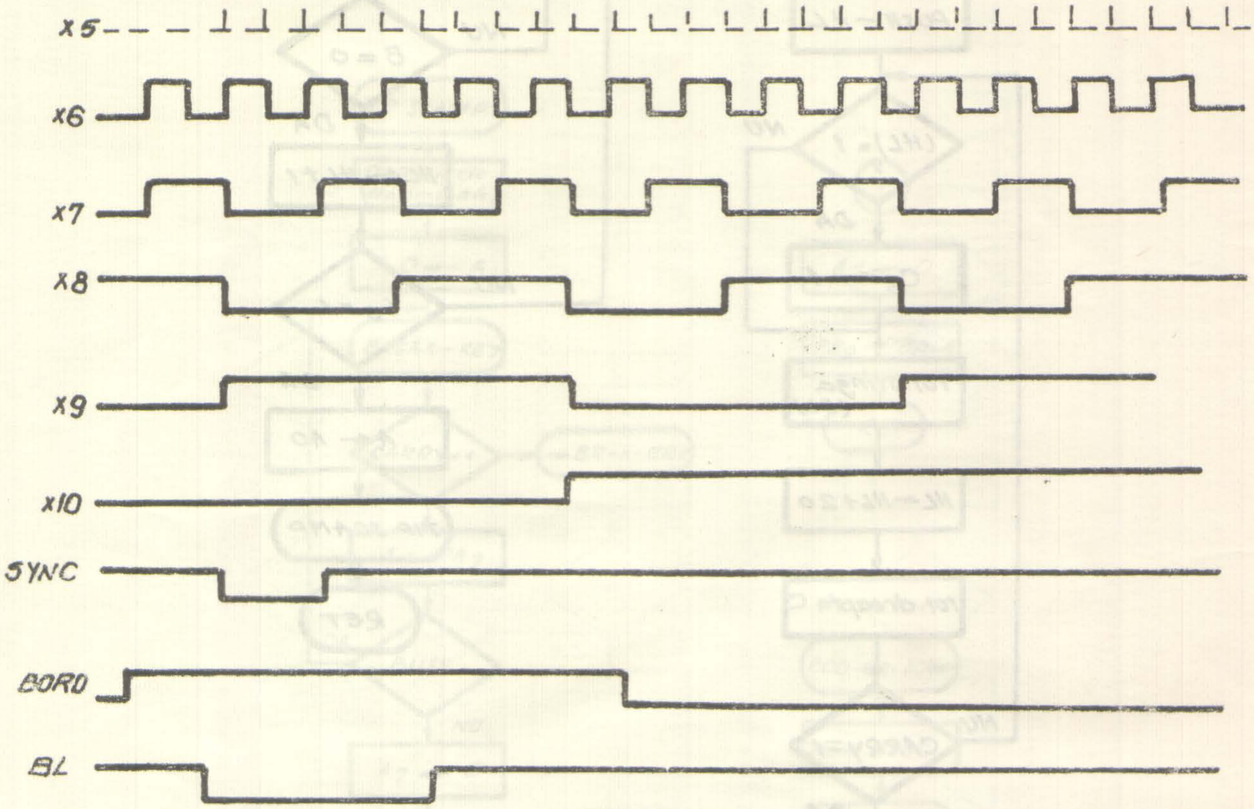


Fig. 4.1.

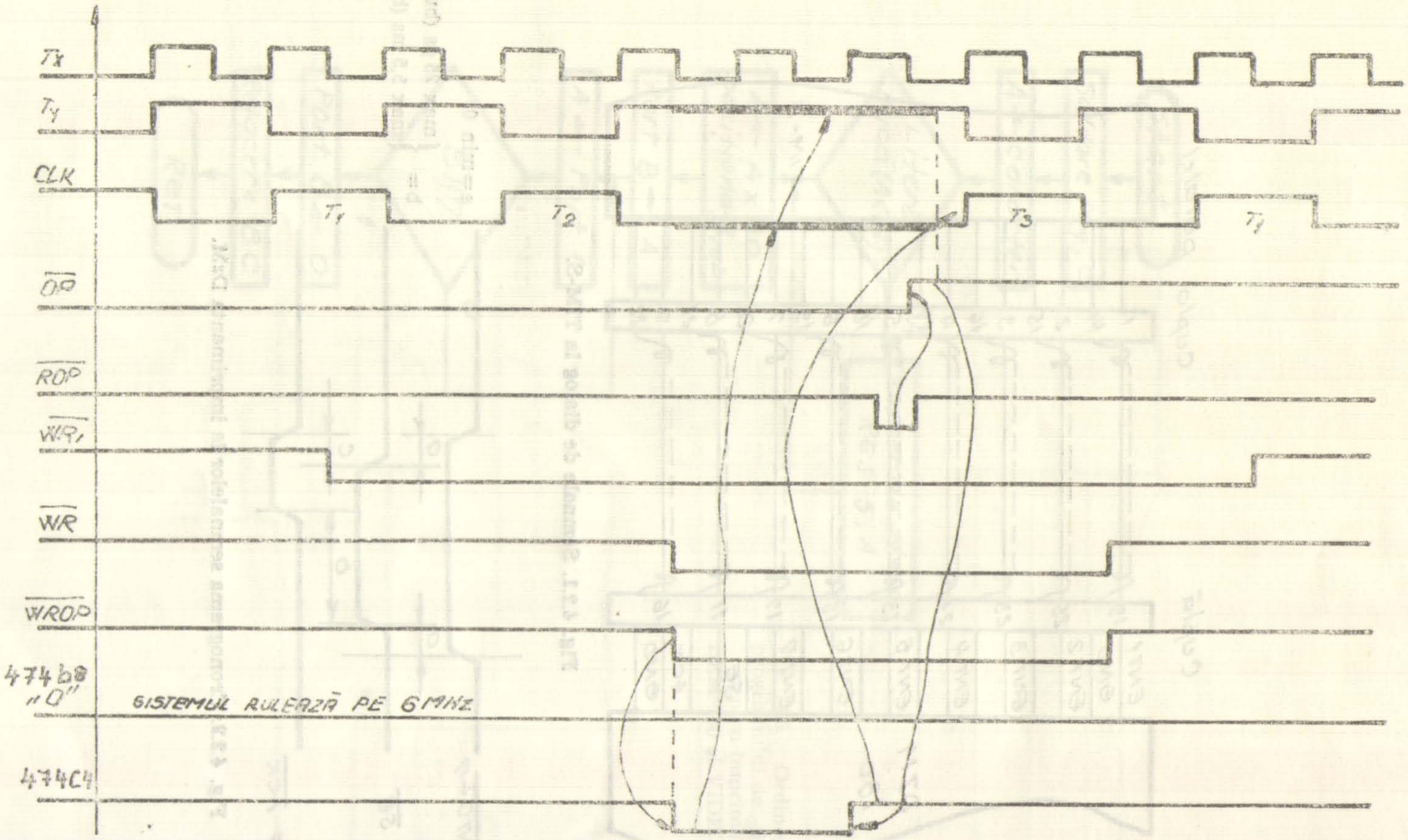


Fig. 4.2. Suspendare tact.

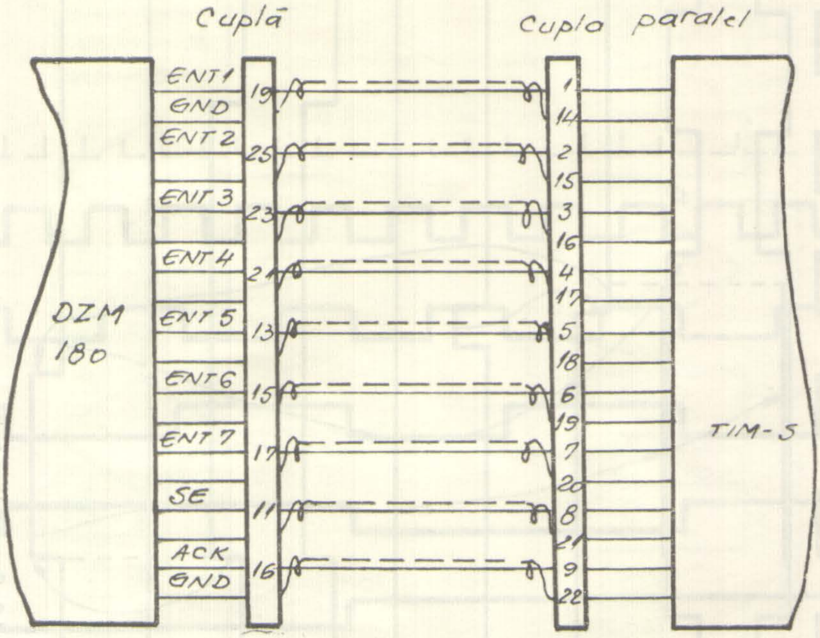


Fig. 4.2.1. Semnale de dialog la TIM-S.

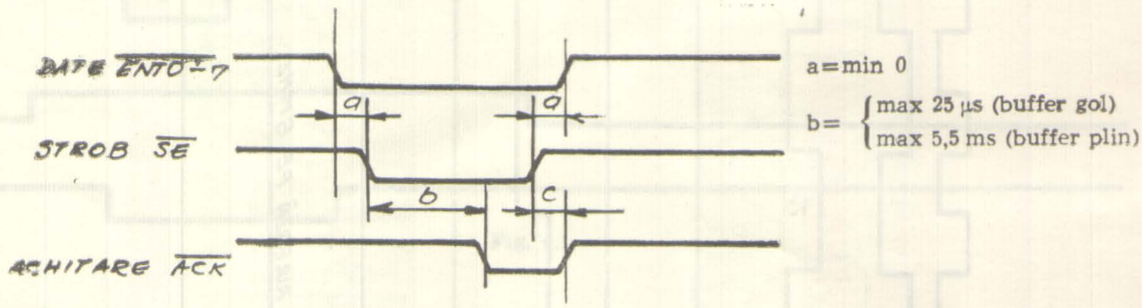


Fig. 4.2.2. Cronograma semnalelor la imprimanta DzM.

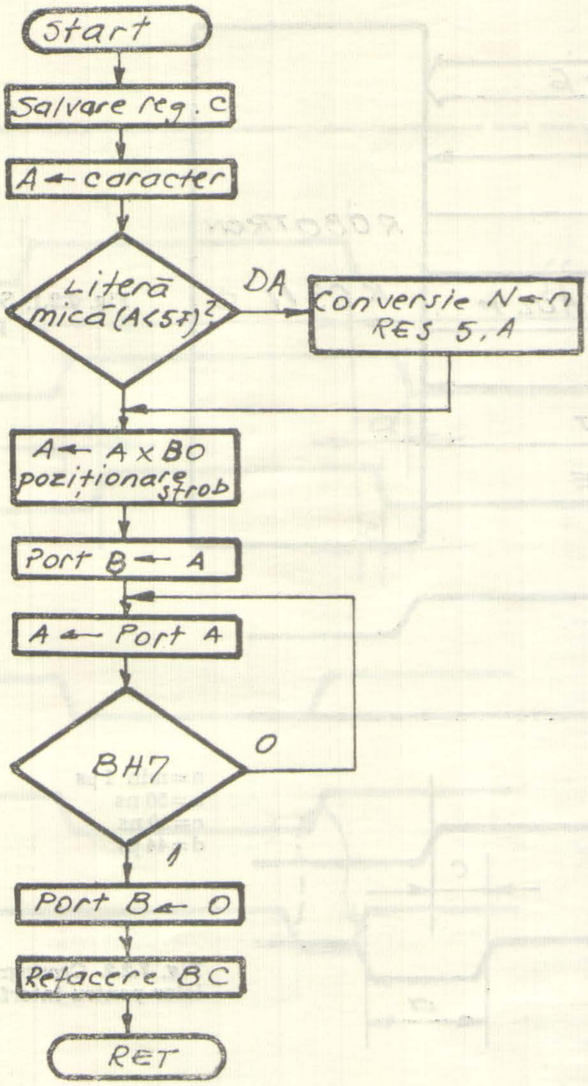


Fig. 4.3. Ordinograma programului de tipărire pentru imprimanta DZM 180 și VIDEOTON.

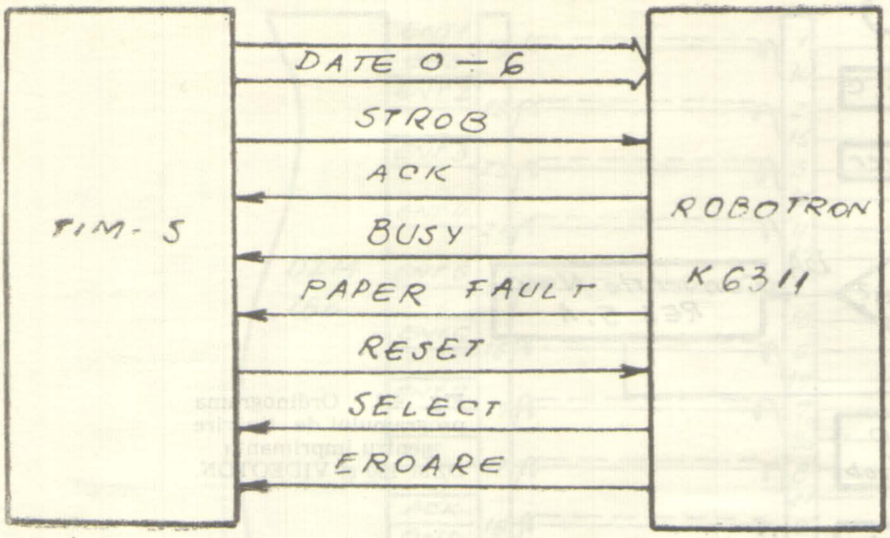
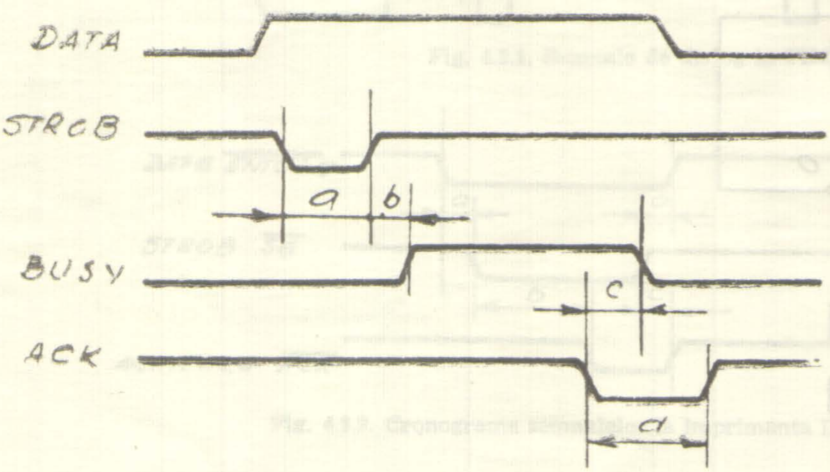
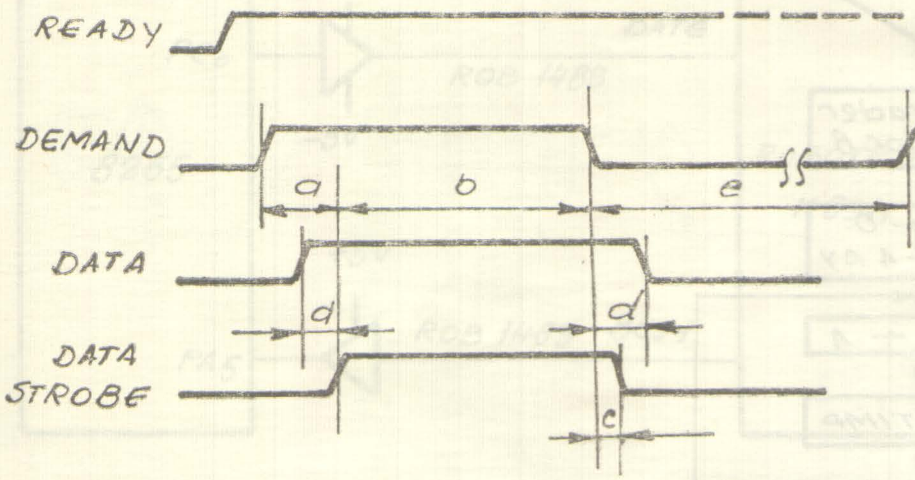


Fig. 7.2.1. Semnalele interfeței paralele.



a = min 1 μs
b = 50 ns
c = 50 ns
d = 44 μs

Fig. 7.2.2. Cronograma semnalelor pentru interfața paralelă.



a = min 0
b = max 1 μs
c = min 0ns, max 200 ns
d = min. 50 ns
e = min 1,5 μs

Fig. 5.2.1.a. Cronograma semnalelor la imprimanta VIDEOTON.

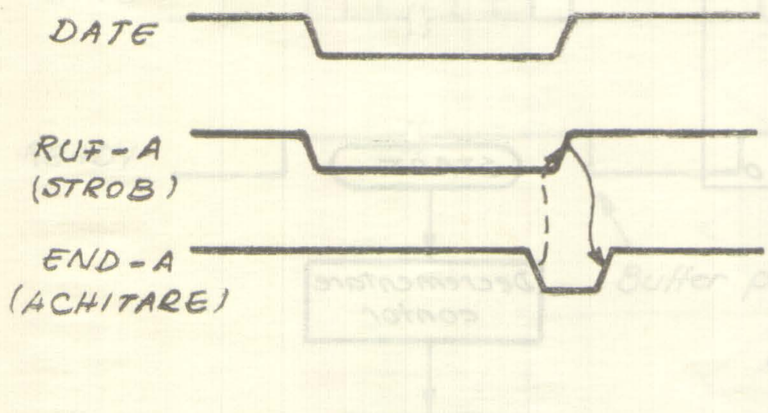


Fig. 5.2.1.b. Protocolul tip VIDEOTON.

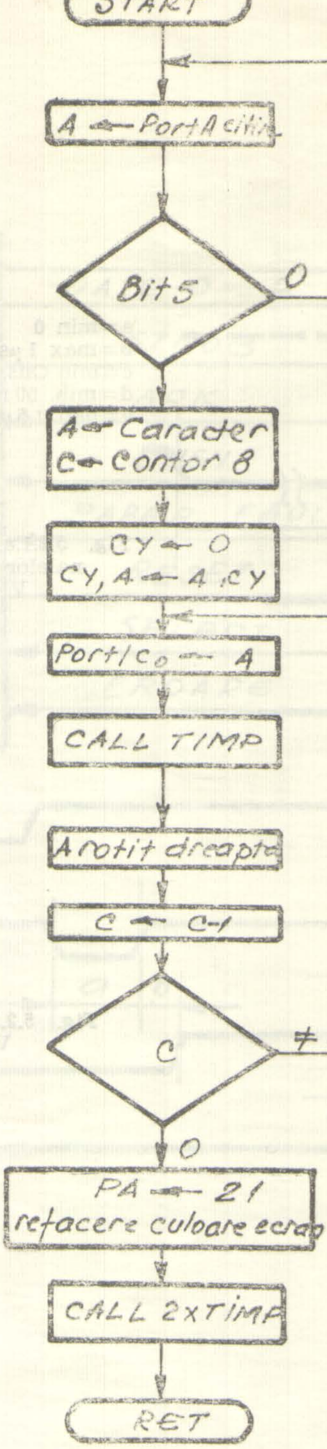


Fig. 6.3. Ordinograma programului de comandă la transmisia serie SERIE.

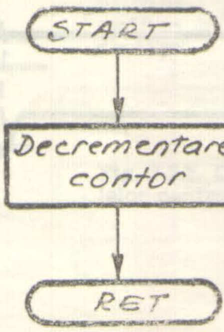


Fig. 6.3.1. Ordinograma subrutinei TIMP.

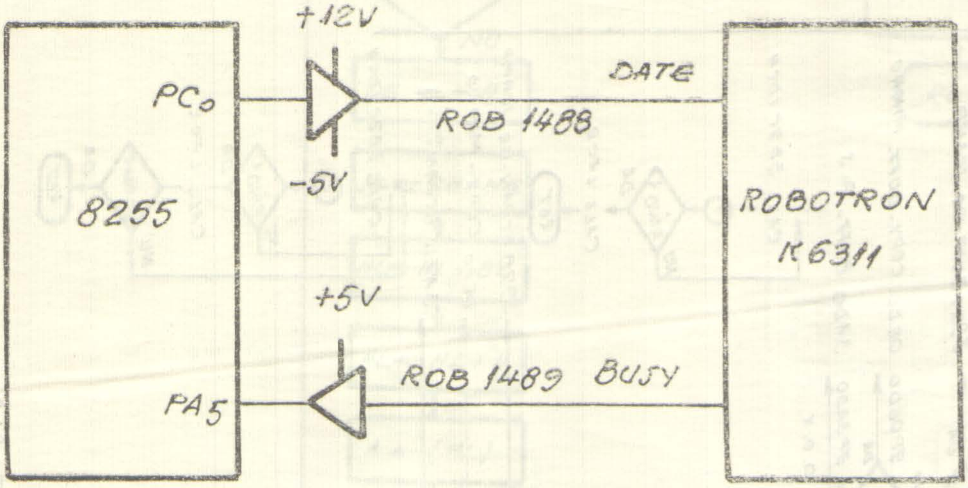


Fig. 7.2.3. Semnalele interfeței serie.

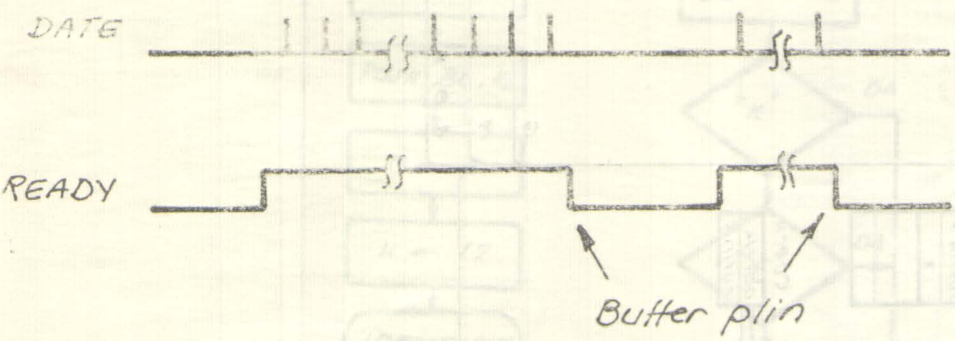


Fig. 7.2.4. Cronograma semnalelor serie.

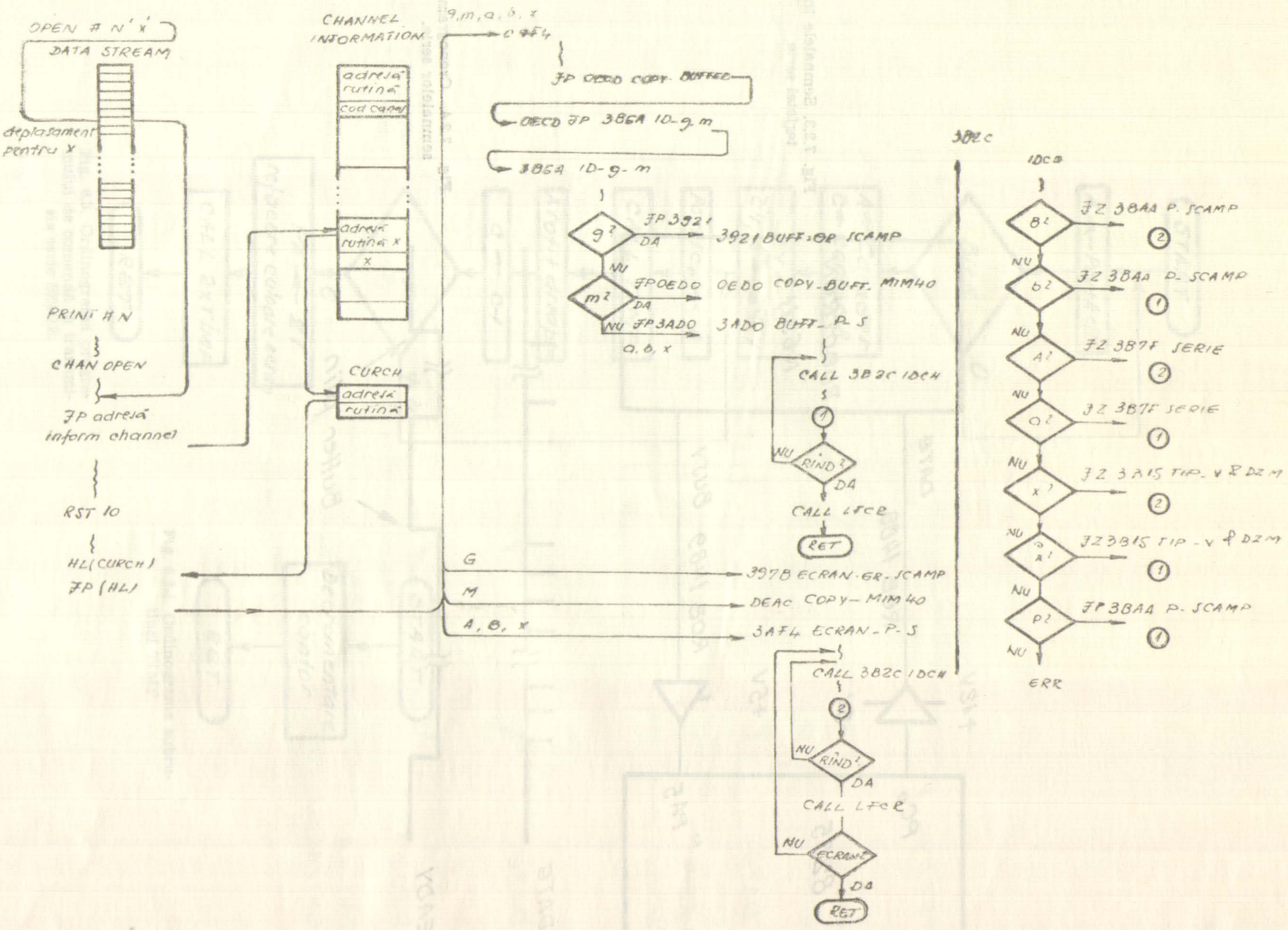

OPEN # N 'x'
DATA STREAM
deplasament pentru x
PRINT # N
CHAN OPEN
JP adresă inform channel
RST 10
HL(CURCH)
JP (HL)
CHANNEL INFORMATION
adresă rutină
cod canal
adresă rutină x
x
CURCH
adresă rutină
g, m, a, b, x
C 9F4
JP OEDO COPY-BUFFER
OECD JP 38EA ID-g-m
38EA ID-g-m
g?
DA
JP 3921
3921 BUFF: GR SCAMP
NU
m?
JPOEDO
OEDO COPY-BUFF. MIM40
NU JP 3ADO
3ADO BUFF-P-S
a, b, x
CALL 3B2C IDCH
RIND?
CALL LFCR
RET
G
397B ECRAN-GR-SCAMP
M
DEAC COPY-MIM 40
A, B, x
3AF4 ECRAN-P-S
CALL 3B2C IDCH
RIND?
CALL LFCR
ECRAN?
RET
3B2C
IDCH
B?
JZ 3BAA P. SCAMP
b?
JZ 3BAA P. SCAMP
A?
JZ 3B7F SERIE
a?
JZ 3B7F SERIE
x?
JZ 3A15 TIP-V & DZM
X?
JZ 3B15 TIP-V & DZM
P?
JP 3BAA P. SCAMP
NU
ERR

Fig. 8A.2.

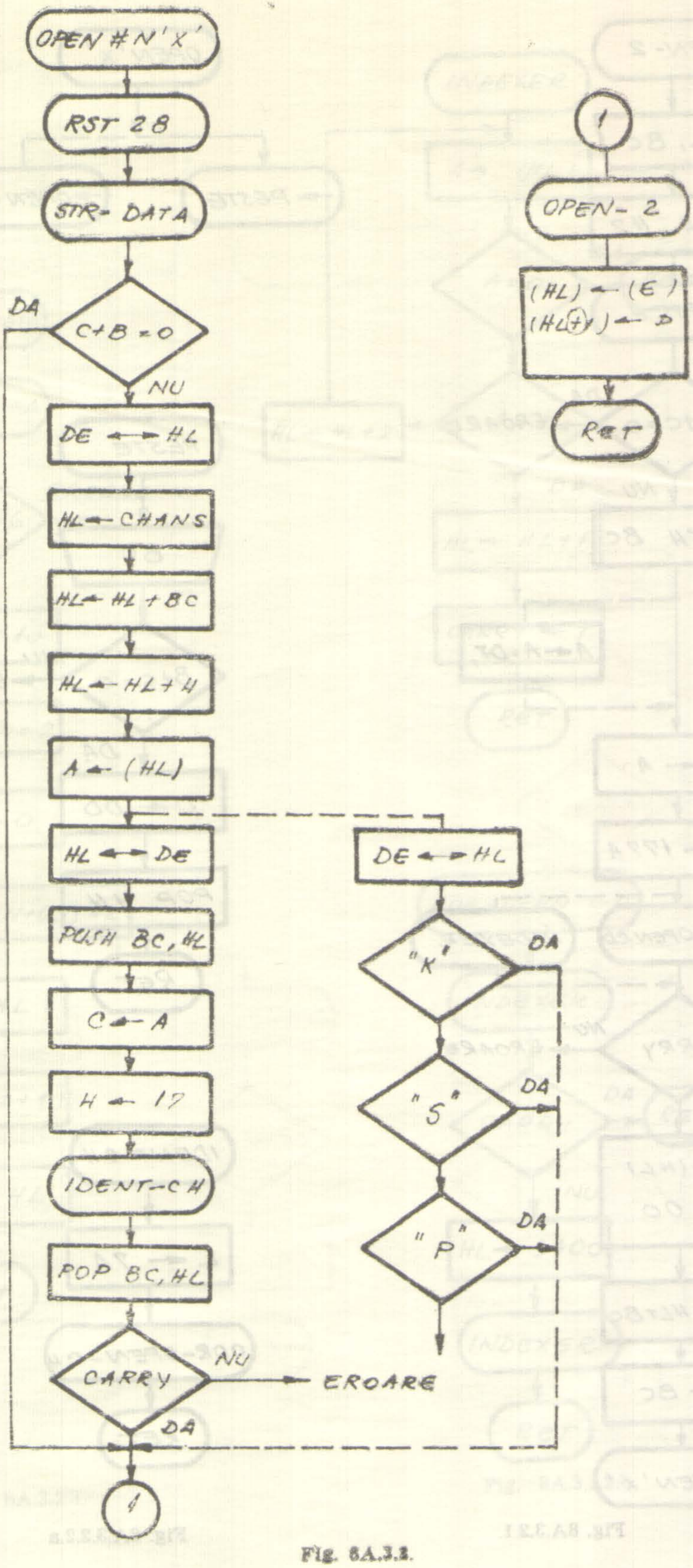

OPEN #N'X'
RST 28
STR-DATA
DA
C+B=0
NU
DE ↔ HL
HL ← CHANS
HL ← HL+BC
HL ← HL+4
A ← (HL)
HL ↔ DE
PUSH BC,HL
C ← A
H ← 17
IDENT-CH
POP BC,HL
CARRY
NU
EROARE
DA
1
DE ↔ HL
"K"
DA
"S"
DA
"P"
DA
1
OPEN-2
(HL) ← (E)
(HL+1) ← D
RET

Fig. 6A.3.2.

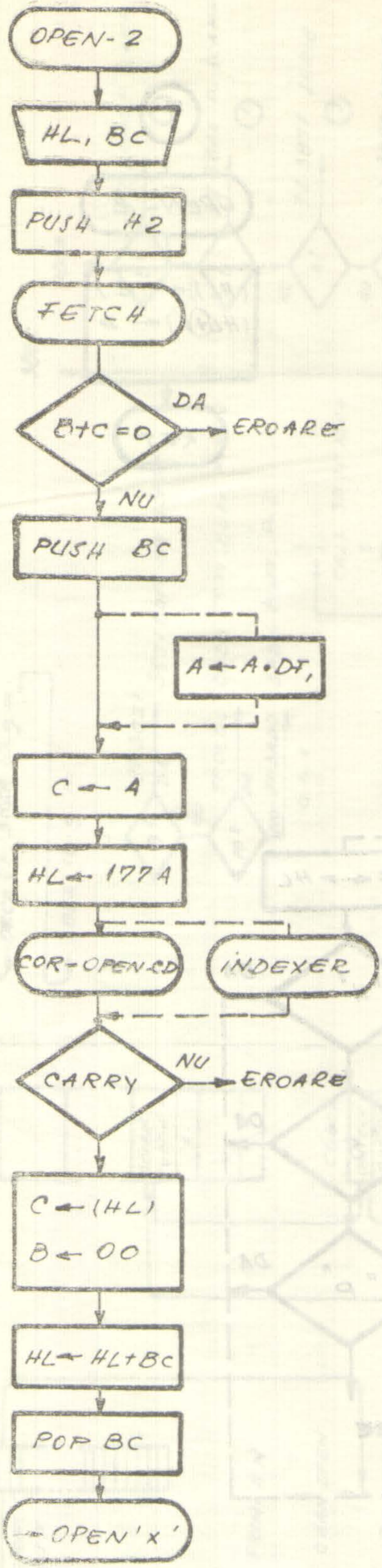


Fig. 8A.3.2.1.

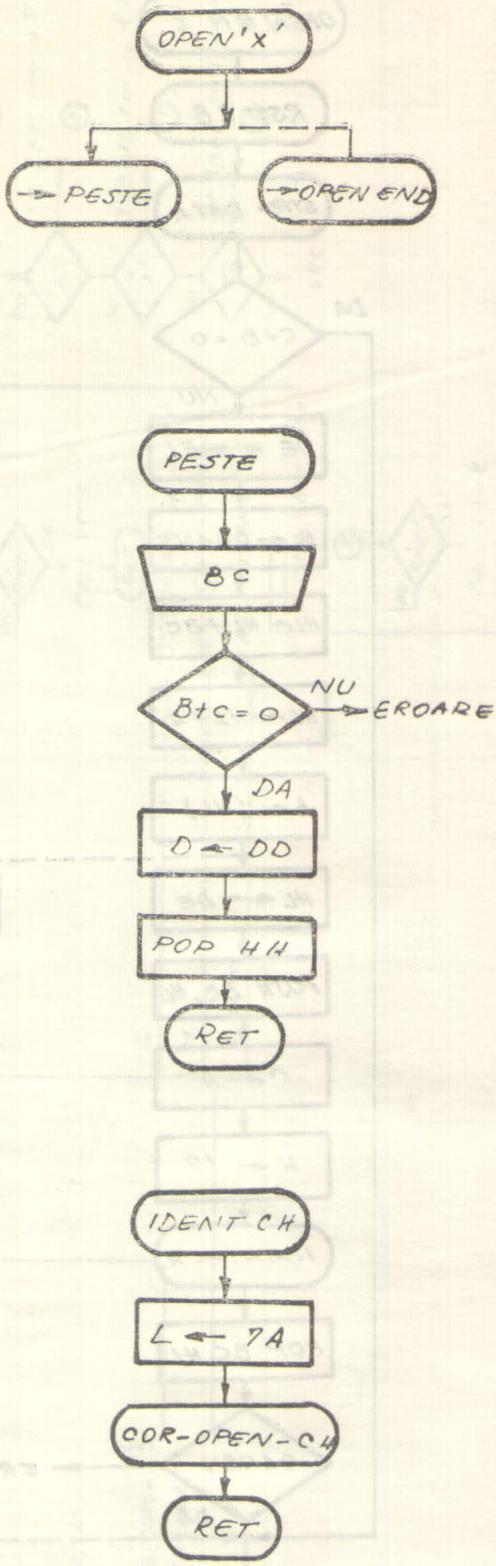


Fig. 8A.3.2.2.a

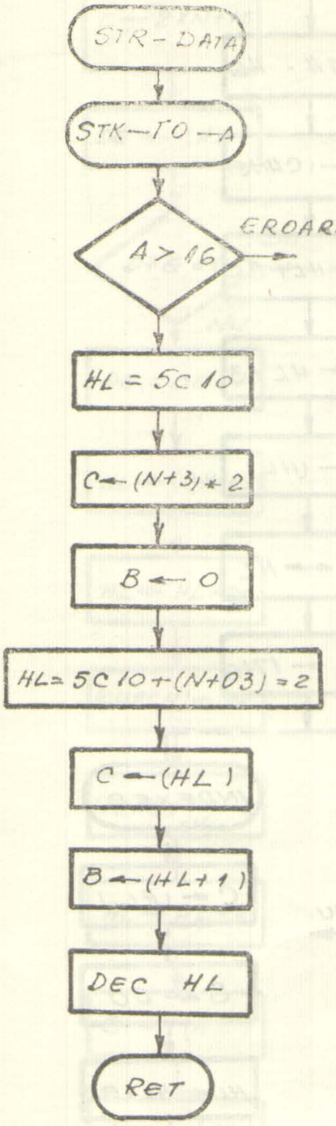


Fig. 8A.3.2.3.

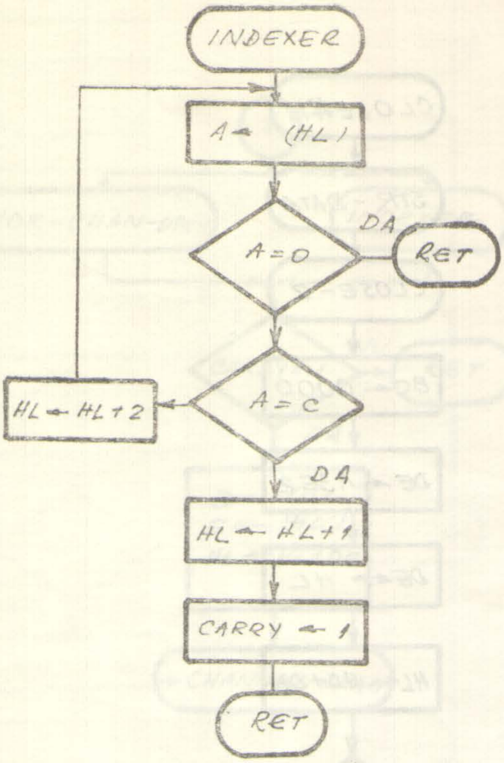


Fig. 8A.3.4.1.

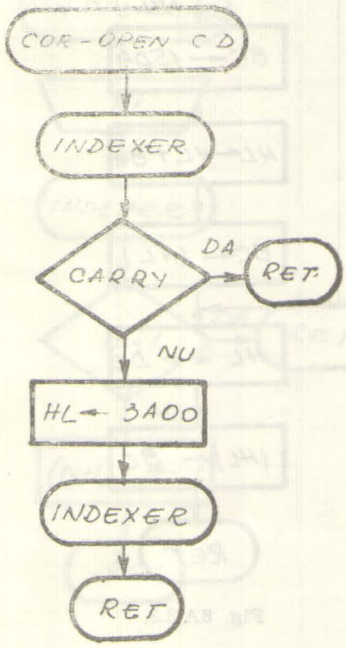


Fig. 8A.3.2.2.b.

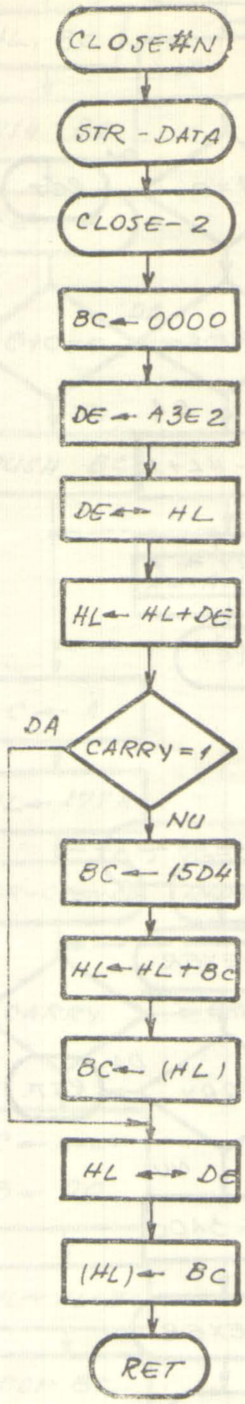


Fig. 8A.3.3.

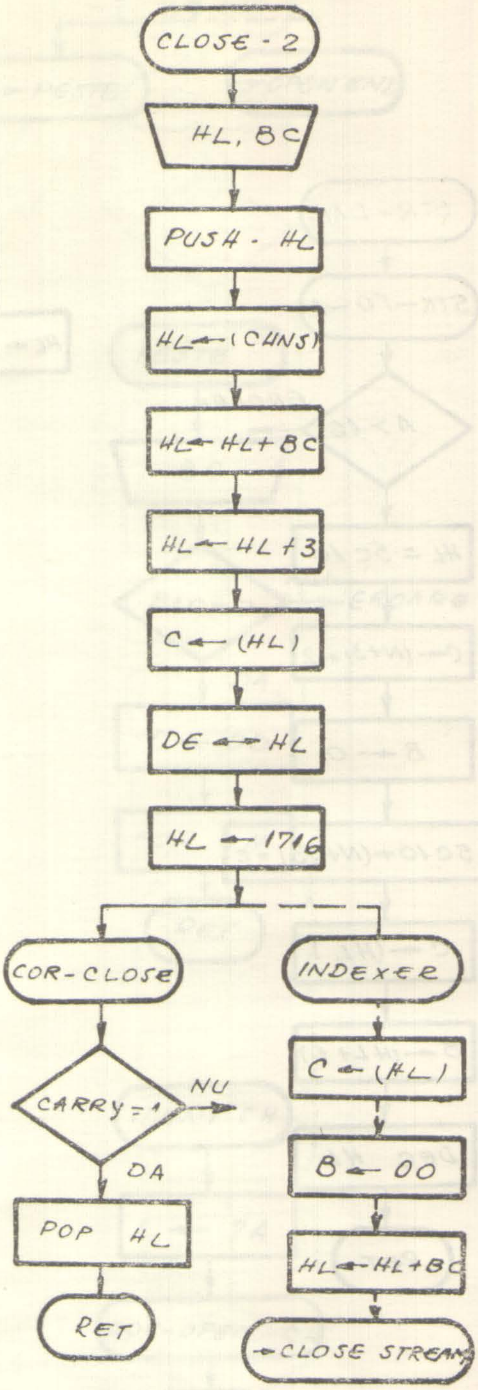


Fig. 8A.3.3.1.

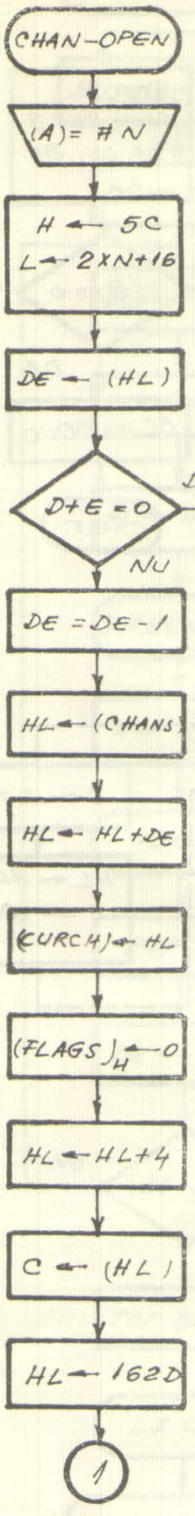


Fig. 8A.3.4.1.

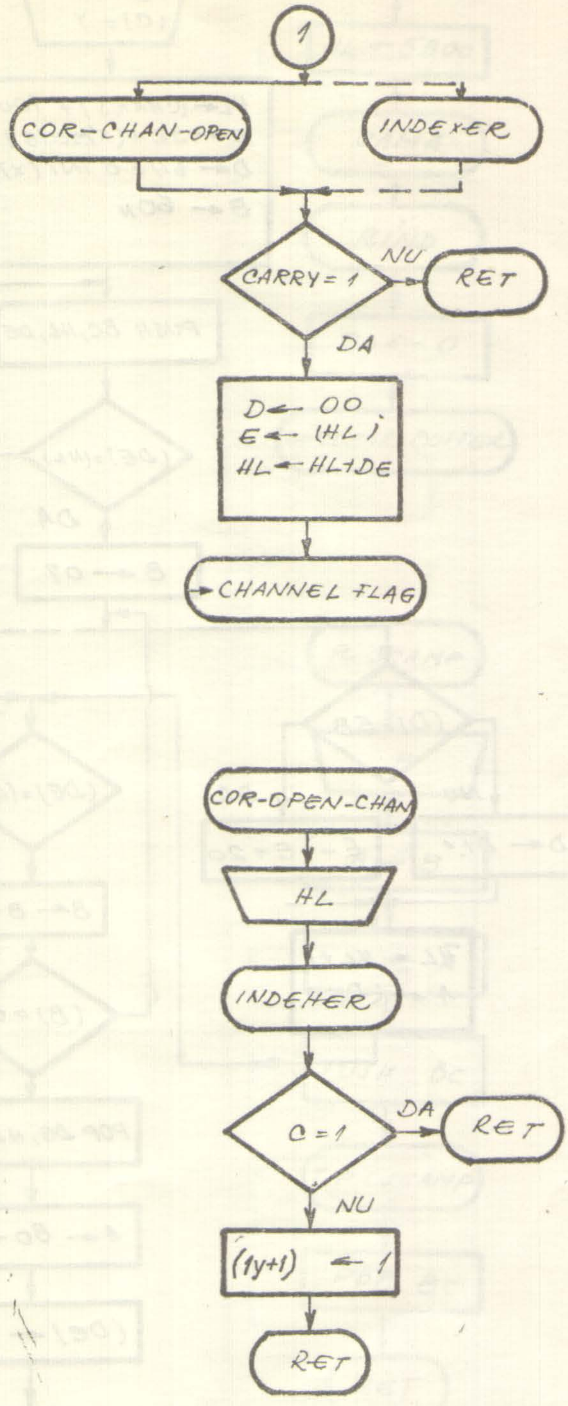


Fig. 8A.3.4.2.

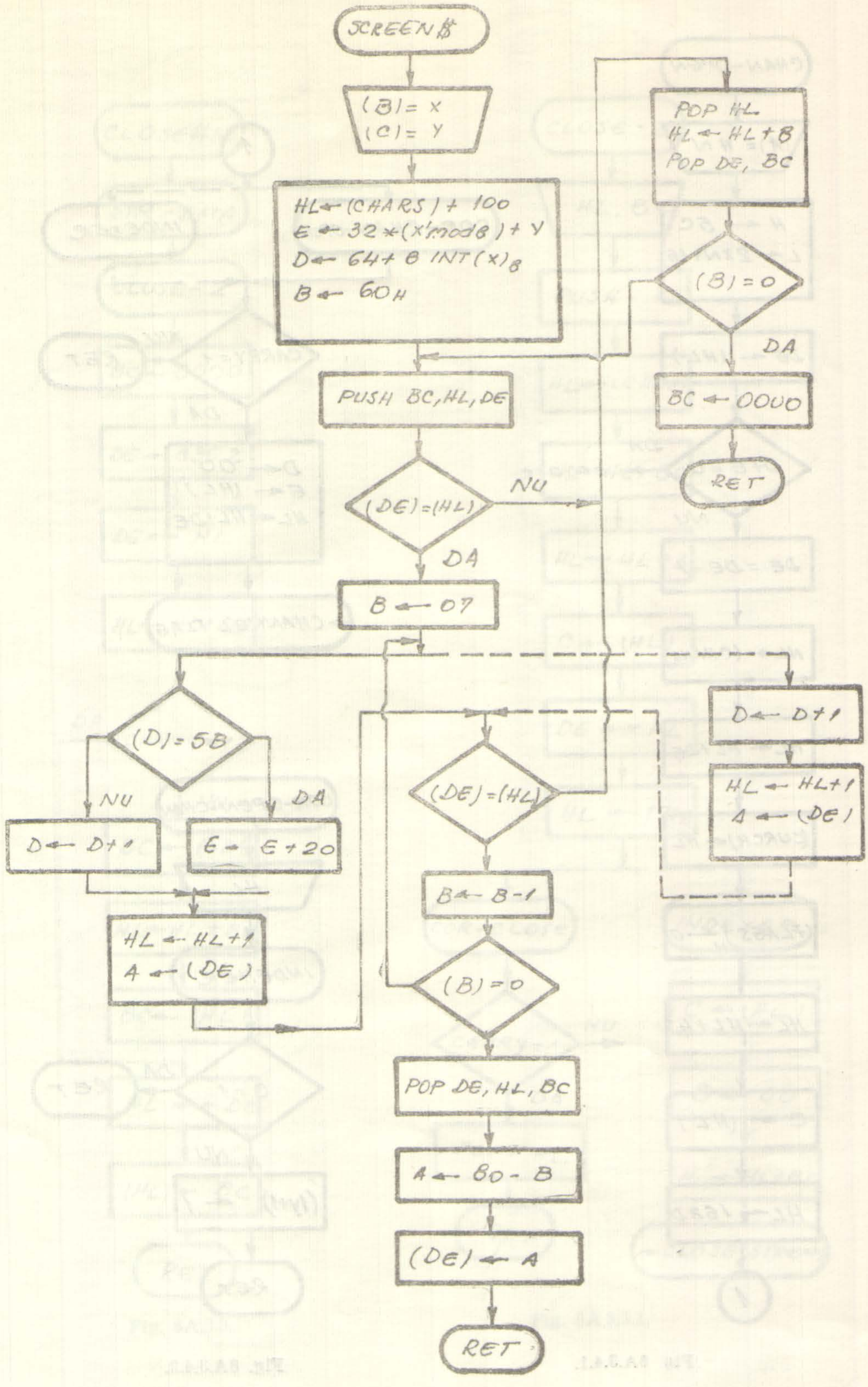

SCREEN$
(B) = X
(C) = Y
HL ← (CHARS) + 100
E ← 32 * (X'mod8) + Y
D ← 64 + 8 INT(X)8
B ← 60H
PUSH BC, HL, DE
(DE) = (HL)
NU
DA
B ← 07
(D) = 5B
NU
DA
D ← D+1
E ← E+20
HL ← HL+1
A ← (DE)
(DE) = (HL)
B ← B-1
(B) = 0
POP DE, HL, BC
A ← 80 - B
(DE) ← A
RET
POP HL
HL ← HL+B
POP DE, BC
(B) = 0
DA
BC ← 0000
RET
D ← D+1
HL ← HL+1
A ← (DE)

Fig. 8A.3.8.

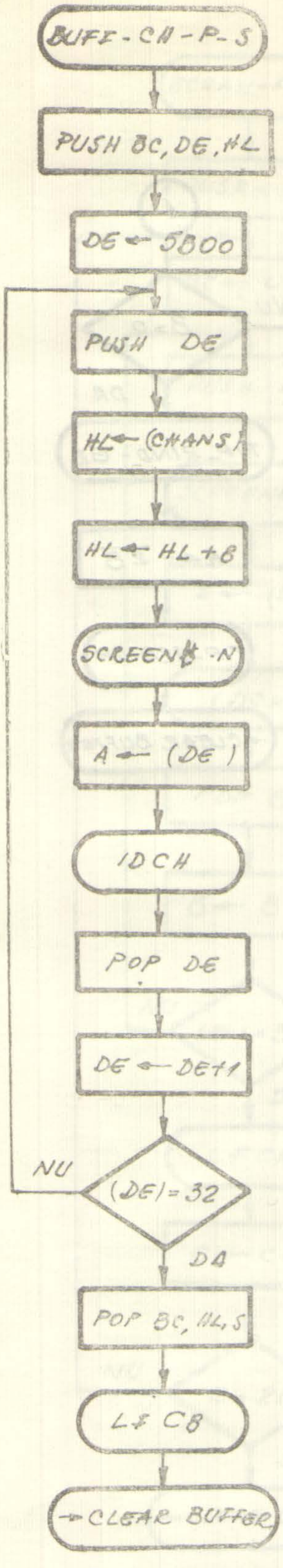


Fig. 8A.3.10.1.

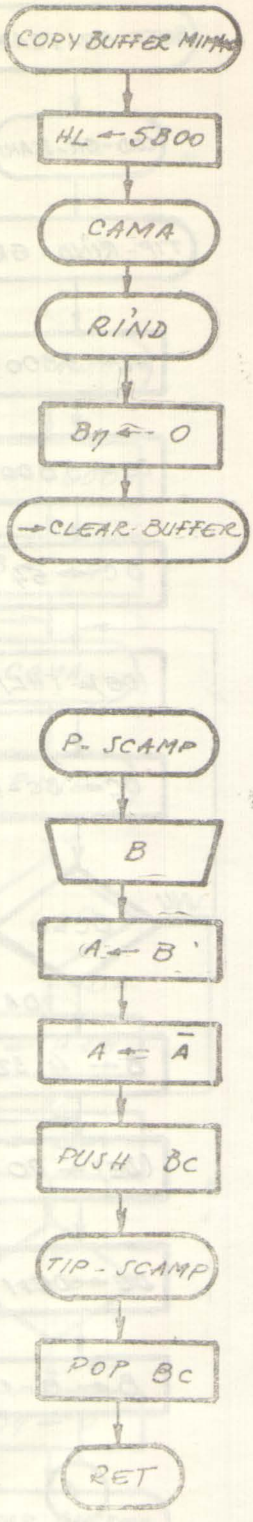


Fig. 8A.3.11.4.

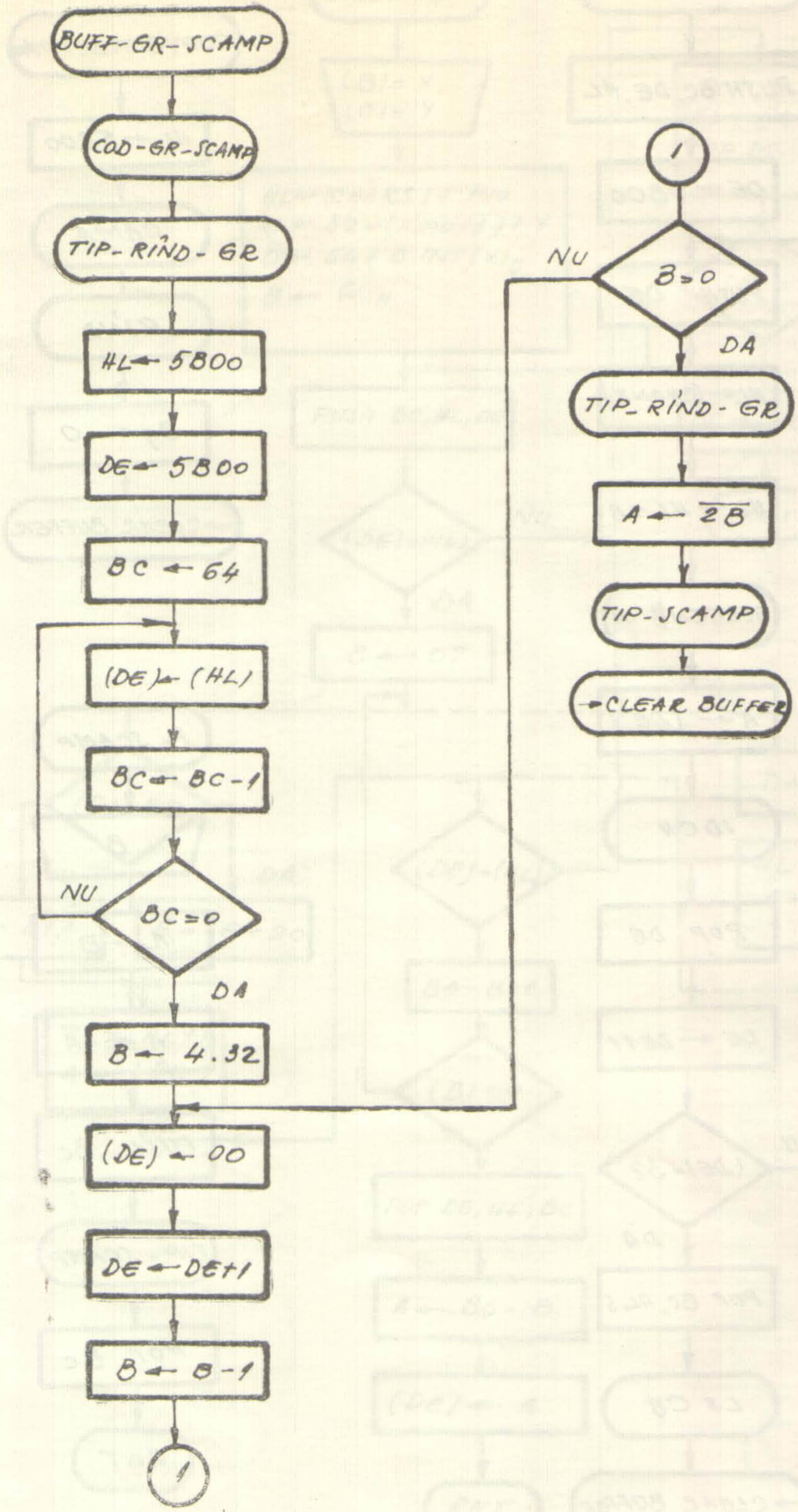

BUFF-GR-SCAMP
COD-GR-SCAMP
TIP-RÎND-GR
HL ← 5800
DE ← 5B00
BC ← 64
(DE) ← (HL)
BC ← BC-1
NU
BC=0
DA
B ← 4.32
(DE) ← 00
DE ← DE+1
B ← B-1
1
1
NU
B=0
DA
TIP_RÎND-GR
A ← 2B
TIP-SCAMP
→CLEAR BUFFER

Fig. 8A.3.10.3.

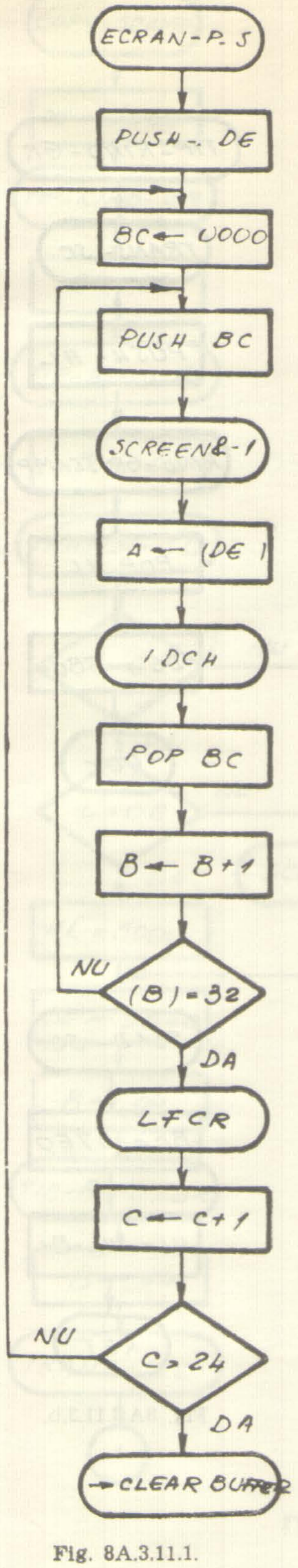


Fig. 8A.3.11.1.

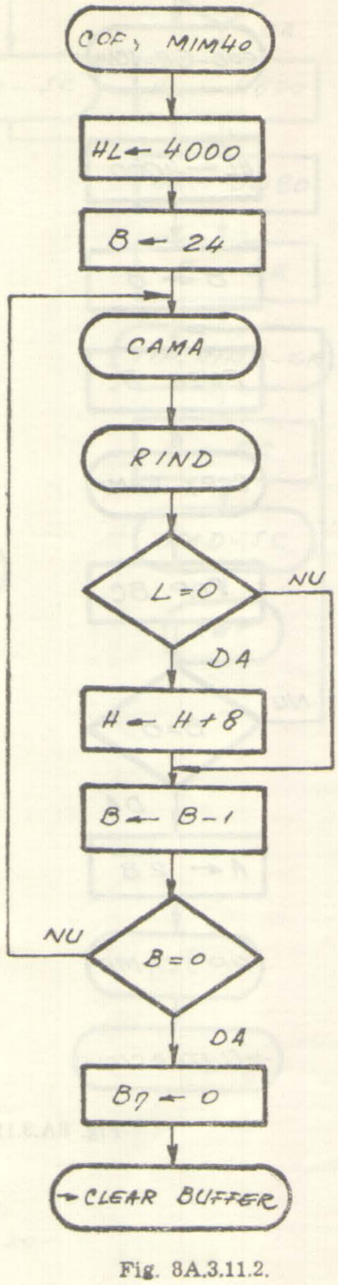


Fig. 8A.3.11.2.

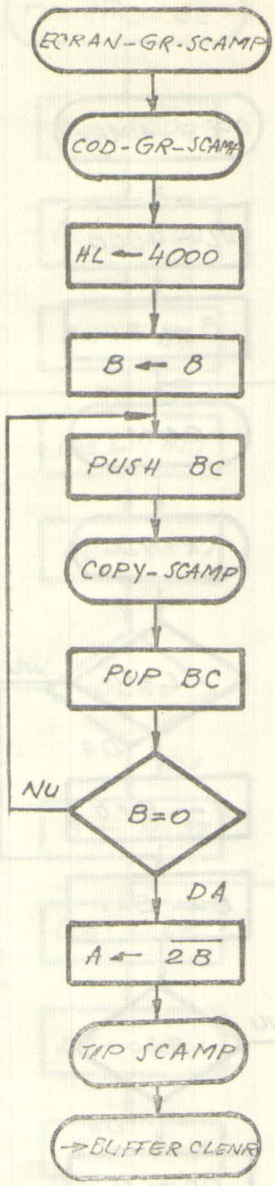


Fig. 8A.3.11.3.a.

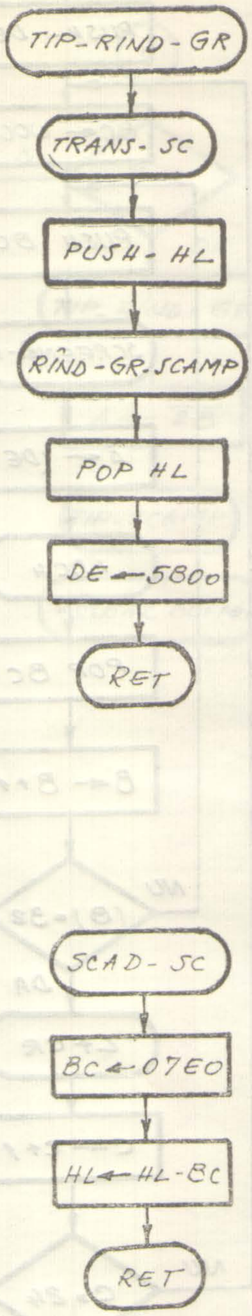


Fig. 8A.3.11.3.b.

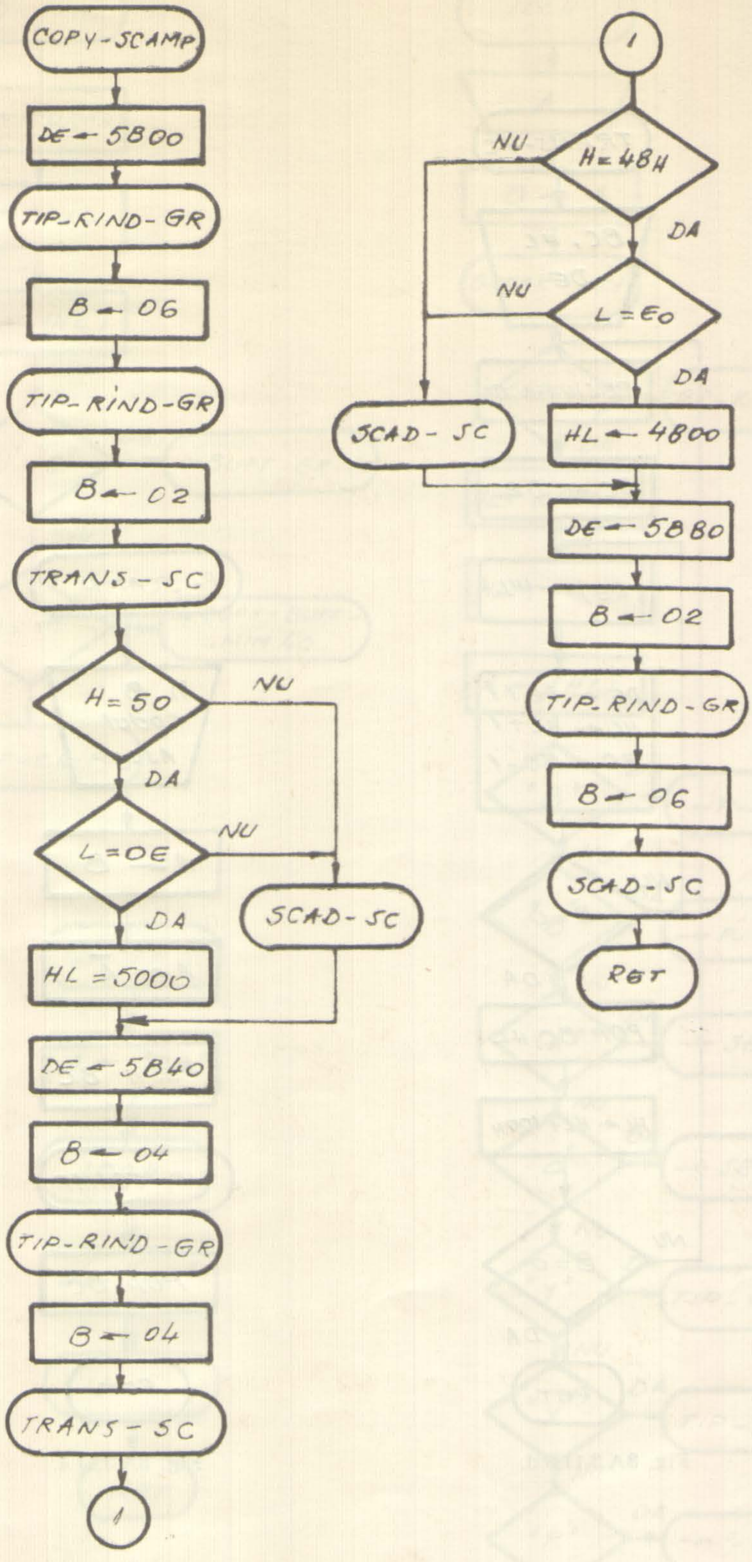

COPY-SCAMP
DE ← 5800
TIP-RIND-GR
B ← 06
TIP-RIND-GR
B ← 02
TRANS-SC
H=50
NU
DA
L=0E
NU
DA
SCAD-SC
HL =5000
DE ← 5B40
B ← 04
TIP-RIND-GR
B ← 04
TRANS-SC
1
1
H=48H
NU
DA
L=E0
NU
DA
SCAD-SC
HL ← 4800
DE ← 5B80
B ← 02
TIP-RIND-GR
B ← 06
SCAD-SC
RET

Fig. 8A.3.11.3.c.

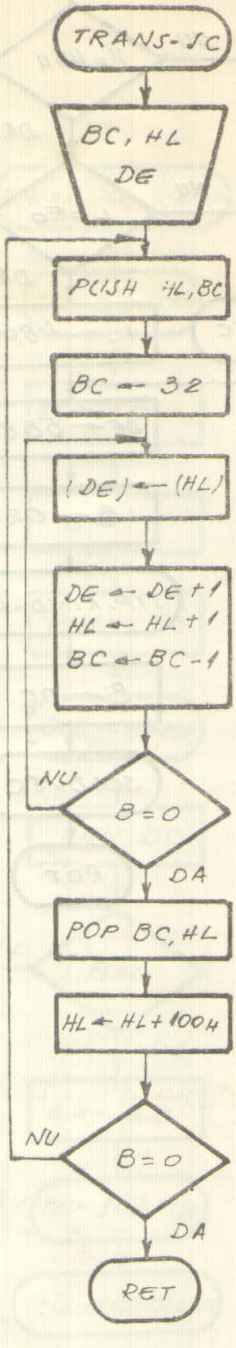


Fig. 8A.3.11.3.d.

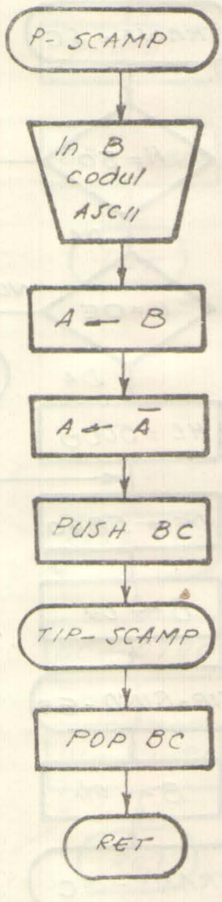


Fig. 8A 3.11.4.

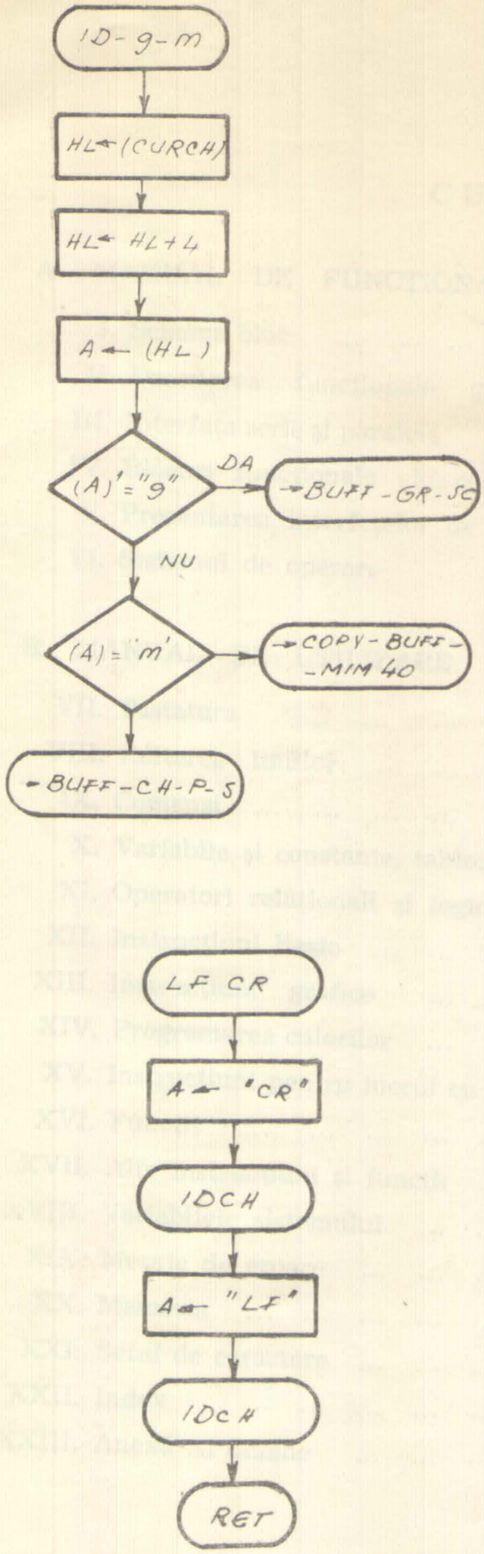


Fig. 8A.3.12.a.

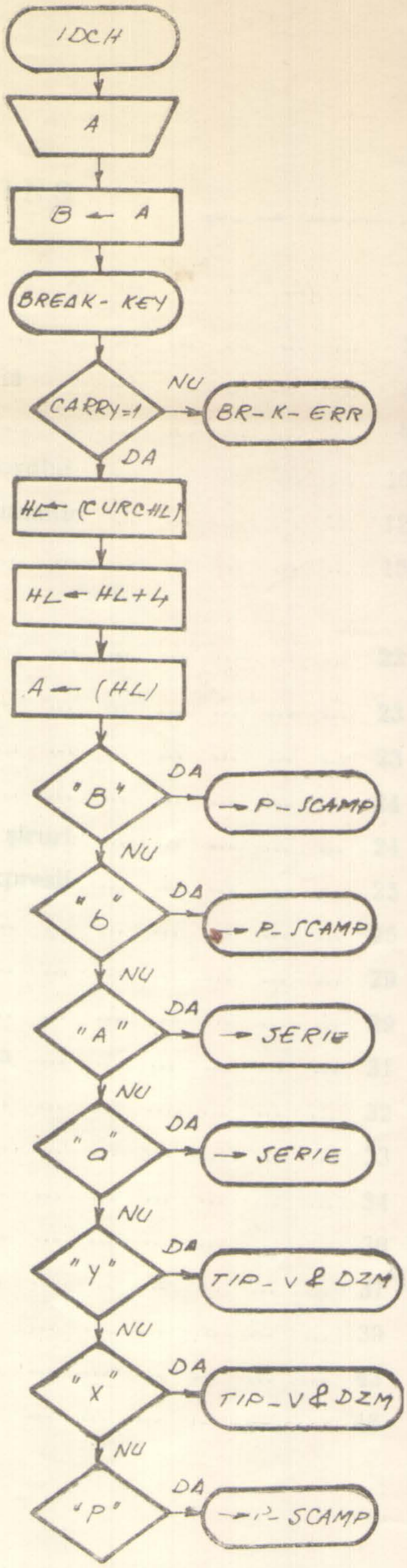


Fig. 8A.3.12.b.

CUPRINS

I.P.B.T. 1792

I.P.B.T. 1792

1900 timișoara str. gh. lazăr 9 tel. 961 30078, telex 43 380